REACTIVE ANT COLONY OPTIMIZATION FOR ROUTING AND ITS CONVERGENCE

By

XU HAOYU

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Master of Philosophy

in

Information Engineering

Acknowledgement

First of all, I would like to express my deep gratitude to my supervisor, Professor Kwang-Mong Sim for his guidance and advice throughout this research work. His impressive idea and outstanding writing skills provide me with excellent support on my research. It is a very valuable experience for me to have discussions with him on different research problems. I would like to thank him for his encouragement and support, as well as his trust and patience, during these years.

With this opportunity, I would also like to thank Prof. Kenneth Shum and Prof. Rong Jae Chen for their suggestions during our group meeting. There are also many thanks to my group members for discussions with me on various topics of communication networks, optimization and queueing theory. These include Mr. Ma Ke, Mr. Chau Chi Kin, Mr. Sun Wen Hong, Mr. Wong Kar Yiu, Miss. Sun Tong, and Miss Wang Shi Yu.

Last but not the least, I would like to thank my family for their endless love and support, especially to my parents.

Abstract

This thesis presents a new ACO routing approach, Reactive Ant Colony Optimization (RACO), for routing in packet-switched networks. Being inspired by the real ants' capability of finding the shortest path, ACO has been applied to solve problems in combinatorial optimization and network routing by modeling ants as a society of mobile agents. While some existing ACO approaches use both forward ants and backward ants causing more routing overhead to the network, or only use forward ants suffering from the limitation in sole application to the symmetric network, RACO eliminates the use of backward ants and is not limited to the symmetric network.

Besides, RACO's rationale is to use the shortest path for routing with low input traffic and as input traffic increases, additional paths are used to avoid overloading the shortest path. This rationale is coherent with the principle of traditional optimal routing. Thus, RACO can be seen as a heuristic method trying to achieve the performance of optimal routing that is inapplicable in real networks because of its complexity. Simulation results show that RACO outperforms both the shortest path algorithm when the input traffic is high and other existing ACO routing approaches no matter the input traffic is high or low. Theoretical analysis shows that RACO can converge to shortest path under some network conditions.

摘　　要

本篇論文提出了一種新的運用于資訊包交換網路中的基於蟻群最優化的路由方法 —— 活性蟻群最優化。受到螞蟻尋找最短路徑的能力的啓發，基於蟻群最優化的方法通過把螞蟻類比成移動代理 (mobile agent) 來解決一些組合優化和網路路由的問題。 然而，一些已存的方法同時使用會增加網路路由複雜度的前進螞蟻和後退螞蟻，要麼雖然只使用前進螞蟻，但只能運用於對稱網絡中。與此對照，活性螞蟻最優化的方法只採用前進螞蟻但可以運用於非對稱網絡中。

另外，活性蟻群最優化路由的原理是在網路負載較輕的情況下，路由演算法只使用最短路徑；隨著網路負載的增加， 路由演算法將使用一些另外的路去避免過度使用最短的路。這個原理和傳統最優化路由的原則是一致的。所以可以看作是一種經驗方法嘗試去擁有最優化的路由的性能， 而最優化路由由於太過複雜在真正的網路中是不實用的。仿真結果表明活性蟻群最優化路由在網路負載較高的時候性能好於最短路徑路由，並且在任何時候都要好於已存的基於蟻群最優化的路由方法。 理論分析證實活性蟻群最優化路由在一定的網路條件下能夠收斂於最短路。

Contents

List of Abbreviations

ACO	Ant Colony Optimization
RACO	Reactive Ant Colony Optimization
DV	Distance Vector
LS	Link State
DE	Devika's approach
RRA	Reactive Routing Ants
RCCA	Reactive Congestion Control Ants

List of Tables

List of Figures

Chapter 1

Introduction

Today's communication network is becoming more and more heterogeneous and complicated, with new kinds of services being introduced, new techniques being implemented and new devices being connected. Network control (i.e. management of the network resources) plays a rudimentary role for utilizing the new transmission, switching and processing technologies to maximize the providers' profits, and meanwhile to meet the users' different demands.

Routing, a policy of building, employing and maintaining routing tables that design the traffic paths between network endpoints, is at the core of the whole network control system. In cooperation with other network control techniques (i.e. admission, flow and congestion control), routing determines the whole network performances in terms of both quality and quantity services the network can afford.

This thesis introduces a new Ant-Colony-Optimization (ACO)-based routing approach in the communication network, Reactive Ant Colony Optimization (RACO). ACO emerged first as a new meta-heuristic approach for solving combinatorial optimization problems. Later on, it was found that ACO could solve other types of optimization problems that can be encoded as "best path" problems. ACO and RACO will be discussed in subsequent sections in great details.

This chapter is organized as follows. Section 1.1 provides some background regarding network routing and Ant Colony Optimization. In section 1.2, the research statement is provided, including the research objectives. The contributions of this research are presented in the section 1.3. Finally, section 1.4 shows the organization of the whole thesis.

1.1 Background

1.1.1 Routing

Network routing plays a more and more critical role in the communication network. World demand and supply of communications networks services are growing exponentially. With the development of transmission technology and media, especially with the application of optical fiber, more and more congestions occur in the router where the routing decisions are made, rather than in the transmission channels. It is the router that becomes the bottleneck of the communication networks. A metaphor of this case is that congestions often occur at the intersections of the highways. Thus, it is significant to devise some new routing techniques to maximize the network performance in terms of the quality and quantity of services it can deliver.

The main task of routing algorithm is to determine the route among network endpoints for the purpose of optimizing network performance. To fulfill this task, routing algorithms usually assume a set of tasks:

(1) To acquire, organize and distribute the information about the data traffics and network status by the coordination among all the nodes in this network;

(2) To utilize these information efficiently for generating reasonable routes

(routing table is another word) among network endpoints, and meanwhile maximizing the performance objectives;

(3) To forward incoming data traffics to their next nodes according to the routing table.

The implementation of these above tasks depends largely on the underlying network switching and transmission technology. Point 3 severs a good example. As to the forwarding paradigm, *packet* and *circuit* switching are the most widely used. In the circuit switching, there is a connection setup phase that reserves the necessary resources for each incoming session the network will accommodate. In this case, all the data packets follow the same path that is determined in the setup phase. Routers are required to keep the state information of each active data session. On the other hand, the packet switching does not have a setup and reservation phase. It does not maintain the state information so data packets may follow different paths. At each intermediate node, each data packet in one session makes its own decisions on which is its next hop for its own destination.

The work presented in this thesis focuses on the packet-switched networks. But intuitively, the techniques used in this work may be applied to the circuit-switched network. It is hoped that the address of this issue can inspire more research in the future.

Different policies of implementing the above three tasks may serve the criterion of classifying the routing algorithms. In terms of utilizing or neglecting the current network information, routing algorithms usually can be classified into two broad categories: *static routing* and *adaptive routing*. In static routing system, the routes taken by a packet are fixed that is only related to its source and destination, without considering the current network state. Thus, the routing table of a static routing system does not change after it is loaded with values when the system starts. The *routing table* is both a local database and a local model of the global network status holding all the information used by the algorithm to make the local forwarding decisions. Static routing cannot achieve a high throughput under a broad variety of input traffic patterns. It is recommended for some simple network or for some networks that do not care efficiency. In contrast, *adaptive routing* refers to a system that can change its routing table over time according to the network status, especially the network congestion. It is more attractive because it can adapt to the quick varying traffic conditions. Despite this merit, it also has some drawbacks, such as the oscillation in selected routes, causing loop routes, and large fluctuations in measured performance[1].

The routing approach presented in this thesis is adaptive in the sense that network status determines the route generation.

In terms of how to acquire, organize, distribute and utilize network information, routing algorithms can also be classified as *centralized routing* and *distributed routing*. In *centralized routing*, a main controller is responsible for gathering the information about the network status, updating all the nodes' routing table, and making every routing decision. Centralized algorithms can only be used in particular cases and for small networks. This kind of algorithm is infeasible because of the long delays due to gathering information about the network status and broadcasting the decisions. To make things worse, centralized systems are not fault-tolerant in the sense that if the main controller is out of order, the whole system will collapse. In distributed routing, there is no main controller. The computation and selection of routes is shared among the network nodes that exchange necessary information. The distributed paradigm is currently used in the majority of network systems.

This routing approach presented in the thesis is distributed.

Any routing approach has to assume these tasks because routing has the following characteristics[2]:

(1) *Intrinsically distributed.* In a routing system, the databases of network status totally distribute over all the network nodes. At each decision node, the information of other parts of the network can be used is not up-to-date because of the propagation delay.

(2) *Stochastic.* Data flow arrivals, users' request and providers' advertisement generation are stochastic.

(3) *Multi-objective.* A good routing algorithm needs to take several performance measures into account. The most common two are *throughput* (bit/sec) and *average end-to-end delay* (sec). The former measures the quantity of service that a network can deliver in a certain amount of time, while the latter measures the quality of service the network can afford. Other performance measures focus on the utilization of routing algorithm on the network resources in terms of memory, bandwidth and processing ability, such as routing overhead. The remaining performance measures

consider the routing algorithm itself, such as simplicity, distribution and scalability.

(4) *Multi-constraint.* The constraints are imposed by users' requests on low-cost, high quality, stable, distributed and heterogeneous services and by provider's easy operation and maximum profit return.

Ant Colony Optimization (ACO), a heuristic method, seems to be a suitable way to solve the routing problem. The next subsection introduces Ant Colony Optimization (ACO).

1.1.2 Ant Colony Optimization

Ant Colony Optimization is inspired by the real ants' foraging behavior and their capability in finding the shortest path between their nest and food location[4][5][6]. Ants with small amount of cognitive capability, limited individual capabilities seem that they are too simple to be helpful for the complex problems, i.e. network routing and Traveling Salesman Problem. However, as they react instinctively and collectively, they exhibit a powerful ability in solving network routing and some difficult combinatorial optimization problems by stigmergy. Stigmergy is a form of indirect communication through the environment, which is generally a control method, and has recently been demonstrated in the domain of collective applications. When a group of ants travel for foods, they always lay some trails, called *pheromone*, which can be detected by other ants and influence their behaviors. This mechanism reinforces other ants to choose the paths that are taken by the majority. It results in the shorter paths accumulating more pheromone than longer paths and finally this group of ants is able to find the shortest path between their nest and food location.

ACO was initially devised as a construction heuristic method for some combinatorial optimization problems. It received a lot of attention and enjoyed a broad range of applications, such as symmetric and asymmetric Traveling Salesman Problem[7][8][9][10][11], sequential ordering problem[12], quadratic assignment problem[13][14][15], vehicle routing problem[16][17],scheduling problems[18] and graph coloring problem[19].

Later on, it was recognized that not only the combinatorial optimization problems, but also any other type of problems that can be encoded as "shortest path" problems are able to be solved using ACO, such as network routing[20][21][22][23][2][24] (Also please refer to chapter 2, section 1 for a survey). In the communication networks, packets traveling in the network between nodes are analogous to the ant's foraging behavior. So, ants are modeled as mobile agents, which are capable of traveling within the networks and laying some pheromone in intermediate links. In order to find the optimal paths between each pair of nodes, a group of ants are emitted. This group of ants chooses their paths based on the pheromone value on the links: the paths with more pheromone will be chosen with higher probability. They also lay some pheromone on these intermediate links that they have traversed. Some time later, the optimal path between each pair of nodes will have the highest pheromone value. Ants are outstandingly adaptive to be able to choose the path with higher pheromone. Moreover, by the indirect communication scheme – pheromone, each ANT is autonomous and works independently of each other but still be able to communicate without any centralized control scheme. Obviously, the merits of adopting ant's foraging behavior to routing problems are its advantage of adaptiveness, independence and distribution.

1.2 Research Statement

The hypothesis of this work is that we can indeed design an adaptive routing algorithm that is simple, distributed, demonstrably efficient, and provably convergent. This thesis demonstrates the hypothesis by advancing a new adaptive routing algorithm, Reactive Ant Colony Optimization (RACO), which is based on previous works on ACO to meet these complex goals. It is shown by a combination of theoretical analysis and detailed packet-level simulation that RACO is simple, distributed, demonstrably efficient, and provably convergent.

Thus, the objectives of this work are three-fold:

1. To design an ACO algorithm for adaptive routing that is simple, distributed, demonstrably efficient, and provably convergent.
2. To demonstrate through packet-level simulation that the algorithm is efficient.
3. To show through theoretical analysis that the algorithm is convergent under some network conditions.

To meet the first goal, RACO is introduced. As an adaptive routing, RACO relies on the information collected by ants – the delay between any pair of nodes. Using this information, RACO can compute a new routing table at any time. Thus, the algorithm endlessly repeats its three stages: collecting network information, computing of a new route, and updating the pheromone table (probability table). The main goal of RACO is trying to minimize the average end-to-end delay and maximize the system throughput when the input traffic is high. The reactive routing ant (RRA) is devised to find the shortest path among nodes in an asymmetric communication network. Another type of ants used in RACO is called reactive congestion control ant (RCCA) whose job is to alleviate the congestion when congestion occurs. One of the most salient characters of RACO is that it does not use the backward ants to collect the network information. On the other hand, backward ants used in AntNet [2][22][23] may bring three drawbacks:

(1) To bring extra traffic load to the network;

(2) To cause late delay in updating routing tables;

(3) To cause less cooperation between ants;

Another character of RACO is that both ants used in RACO are reactive to network status. RRA is able to update the routing table according to the delays between any pair of nodes. RCCA adjusts the routing table with respect to the loads of links where congestion happens.

To meet the second goal, an experimental testbed has designed been for evaluation. The testbed is implemented in NS[25], a well-known discrete event simulator in the networking research field. The testbed includes some protocols in transporter layer and application layer that make the testbed more realistic.

This thesis meets the third goal through developing a succession of theoretical results. The convergence property of RACO is analyzed. It is shown that for RACO, following results hold:

(1) When the shortest, unidirectional path from one node to another and the

shortest, unidirectional path connecting the same pair of nodes (in the reverse direction) belong to the same bi-directional path, the RACO approach converges to the shortest path; (2) When shortest, unidirectional path from one node to another and the shortest, unidirectional path connecting the same pair of nodes (in the reverse direction) do not belong to the same bi-directional path, the modified RACO approach converge to the optimal paths; (3) The pheromone update rate plays an essential role in the convergence property of RACO approach.

In summary, through a combination of algorithm design, comparative evaluation through packet-level simulation, and theoretical analysis, RACO is shown to be simple, distributed, demonstrably efficient, and provably convergent under some network conditions.

1.3 Main Contributions

The contributions of this thesis can be summarized as follows:

(1) A survey about previous ACO routing algorithms (Chapter 2, Section 1);

(2) A comparative study of ACO routing algorithms with shortest path routing algorithms (Chapter 2, Section 2); a comparative study of ACO routing with optimal routing (Chapter 2, Section 3) that gives researchers a brand new perspective to study ACO based routing algorithms.

(3) A new ACO routing approach, RACO, has been designed (Chapter 3). This approach eliminates the use of backward ants and is still applicable in asymmetric networks. Besides, it also can be seen as a heuristic method to approximate the optimal routing.

(4) An experimental testbed of packet-level simulation has been implemented to comparative evaluation of RACO with one representative of shortest path routing algorithms and two previous ACO routing approaches (Chapter 4).

(5) The convergence property of RACO has been analyzed. This work is the first attempt to analyze the ACO approaches for routing in asymmetric routing. (Chapter 5).

1.4 Thesis Organization

This thesis is organized as follows. Chapter 2 describes the state of current research on ACO routing algorithms, gives a comparative study of ACO routing algorithms with traditional shortest path routing algorithms, and with some previous optimal routing algorithms. A new ACO routing algorithm, RACO, is introduced in chapter 3. Chapter 3 also presents the reasons behind each of its major design ideas. In chapter 4, simulation results of RACO and an evaluation with some other previous work are demonstrated. Chapter 5 analyzes the convergence behavior of RACO in a great detail and reveals that the pheromone update rate plays a critical role in the convergence of RACO routing algorithm. Chapter 6 gives a conclusion to this thesis and outlines areas for further research.

Chapter 2

Ant Colony Optimization Routing

In recent several years, the application of ACO to network routing has enjoyed a remarkable popularity in literatures[2][20][21][22][23][24]. To understand this trend, it seems necessary to have a close look at these ACO based routing algorithms. How these algorithms fulfill the task of routing? What are the differences between ACO routing algorithms and traditional shortest path routing algorithms? What are the differences between ACO routing algorithms and other optimization based routing algorithms (optimal routing)? What are the advantages of ACO based routing?

This chapter describes the state of current research on ACO routing algorithms, gives a comparative study of ACO routing algorithms with tradition shortest path routing algorithms, and with other optimization based routing algorithms. Section 2.1 surveys most of important ACO algorithms by the application domain in terms of circuit-switched networks and packet-switched networks. Section 2.2 discusses the differences between ACO routing and traditional shortest path routing algorithms/protocols such as the Bellman-Ford algorithm[26] (another name is Distance Vector algorithm) on which RIP[27] is based and Dijkstra algorithm[28] that is used by OSPF[29] to compute the shortest path. The comparative study of the difference between ACO routing and optimal routing is demonstrated in section 2.3.

2.1 ACO Routing Algorithms

This section surveys most of existing ACO algorithms that are divided into two groups based on their application domains, i.e. circuit-switched or packet-switched networks.

2.1.1 ACO Routing Algorithms in Circuit-Switched Networks

Schoonderwoerd *et al*'s [20][21][30][31][32][33] *Ant-Based System* was devised to routing and load-balancing problem in circuit-switched networks. His work was like Appleby and Steward's load management agents[34] in some extent.

Schoonderwoerd's approach has following characteristics:

- *Pheromone tables.* Tables of probabilities that are called "pheromone tables", as the pheromone strengths are represented by these probabilities, replace the routing tables in this approach. Every node has a pheromone table for every possible destination in the network, and each table has an entry for every neighbor. Please refer to the Fig 2.1.
- *Symmetric link cost.* The costs of any directed links are equal bi-directionally.
- *Simple Updating Rules.* When an ant arrives at a node, the entry in the pheromone table corresponding to the node from which the ant has just come is increased according to the following formula:

  $p = \frac{p_{old} + \Delta p}{1 + \Delta p}$. Here p is the new probability and Δp is the probability (or pheromone) updating rate. The other entries in the table are decreased according to: $p = \frac{p_{old}}{1+\Delta p}$, where $\Delta p = \frac{d}{age} + c$, c and d are constants.

- *Ageing and delaying ants.* As the above equation shows, Δp reduces progressively with the aging of ants. The consequence of this is to reduce the older ants' influence on pheromone table.
- *Deterministic routing:* When a call generates at a particular node to a destination, the largest probability in the routing table for this destination is examined. The neighboring node corresponding to this probability will be the next node on the route to the destination.
- *Noised added:* Noise means the random walk of the ants that ensures even apparently useless routes are selected occasionally.

In Schoonderwoerd's approach, incoming ants only updates entries in the pheromone table corresponding to their source node. For example, see Fig 2.1, there is an ant moving from node 0 to node 3. It only updates the entries in node 1, node 2 and node 3 corresponding to node 0. Here, this approach makes an assumption that the work is symmetric.

Some additional mechanisms in this algorithm are devised to achieve the goal of load balancing. These mechanisms are *congestion dependent delay*, *increasing influence for younger ants* (*or aging*) and *noise*. The delay temporarily reduces the flow rate of ants from the congested node to its neighbors, thereby preventing those ants increasing the pheromone tables that route the ants to the congested node. By aging, it refers to younger ants' Δp is relatively large so that their effects to the new probability p are more obvious than the older ones. Noise enables ants to choose a path randomly not taking the pheromone table into account. It is one kind of techniques that increase the possibility of exploration that may be help for solving so-called "stagnation" problems.

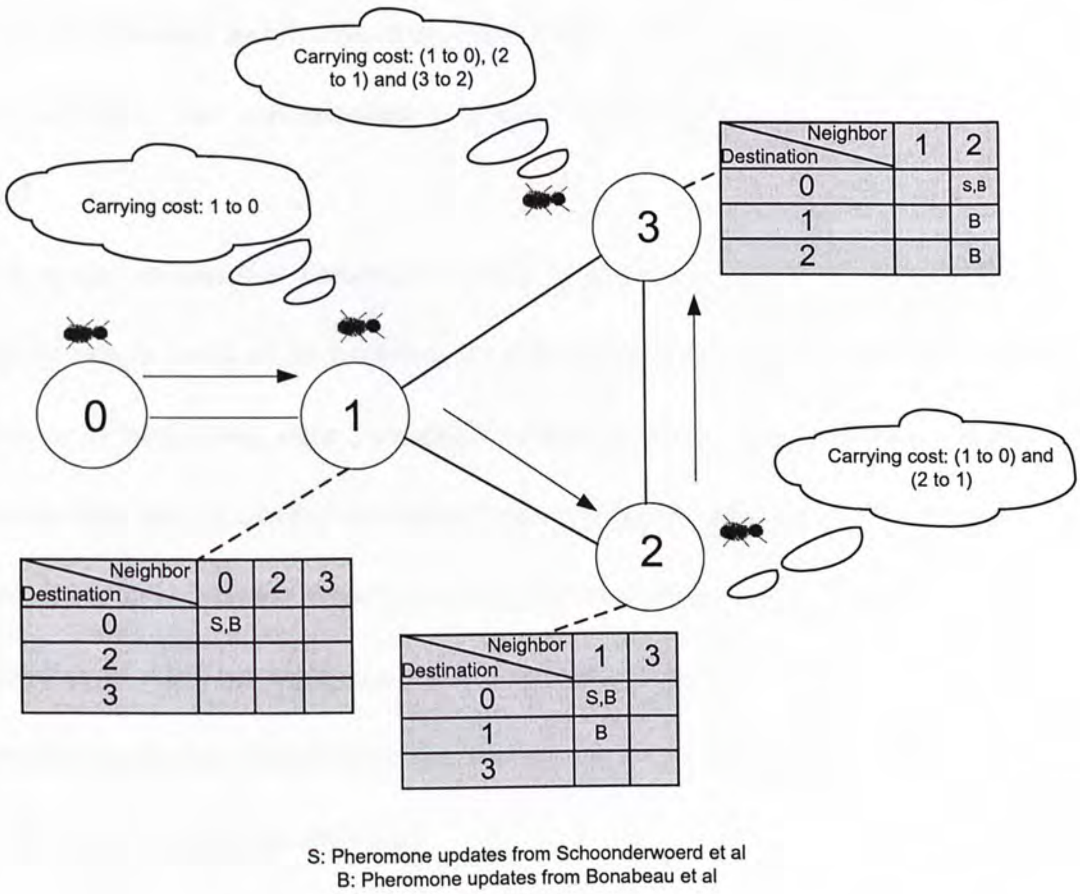

Fig 2.1: The pheromone update policy of Schoonderwoerd et aland Bonabeau et al.

The performance of this approach is measured through the number of dropped calls and the time to adapt to changes in topology and call probability of nodes. Schoonderwoerd's ABS system showed a better performance than Appleby and Steward's system[34] from the experimental results.

Bonabeau et al [35] add an extra feature to Schoonderwoerd's ABS system. While an ant in ABS updates only the entry corresponding to the source node in the pheromone table of each node it passed, Bonabeau *et al*'s [35] smart ants update the pheromone table at each node, *all* entries corresponding to *every* node they pass. For example, see Fig 2.1, when an ant moves from source node 0 to destination 3 through intermediate nodes 1 and 2. After it arrives at node 2, it will update both entry in node 2's pheromone table corresponding to nodes it has passed so far, i.e. node 0 and node1.

It seems obvious that Bonabeau *et al*'s [35] smart ants inherit the weakness of ABS system in terms of its limitation for only applicable in symmetric network. In addition, by performing more pheromone updates at every intermediate node, smart ants are more complex and perform more computation than ants in ABS. On the other hand, fewer smart ants are needed to achieve the same routing results. Using the same testbed as in ABS but augmented with smart ants, Bonabeau *et al* achieved more favorable results than Schoonderwoerd *et al* both in terms of the number of drop calls and the time to adapt network status.

2.1.2 ACO Routing Approaches in Packet-Switched Networks

There are more literatures regarding ACO routing algorithms in packet-switched networks. These works are sub-sectioned roughly according to their publishing year.

2.1.2.1 Devika et al's approach

This approach can be seen as a direct extension of Schoonderwoerd's ABS approach except two things are different. First, this approach is supposed to be used in the packet-switched networks with the symmetric network assumption. Second, instead of using deterministic routing, this approach uses probabilistic routing. Probabilistic routing can be demonstrated as follows: Table 2.1 shows the routing table of node 2 in the network (Fig 2.1). When a packet arrives to node 2 with the destination 3, it will choose node 1 as its next node with probability 0.05, and choose node 3 with probability 0.95.

Table 2.1 Probabilistic Routing Table

Neighbor / *Destination*	*1*	*3*
0	*0.55*	*0.45*
1	*0.95*	*0.05*
3	*0.05*	*0.95*

While deterministic routing chooses the next node with the maximum value of its corresponding entry. Probabilistic routing intrinsically gives us a way of load balancing and a scheme of multi-path routing. Devika *et al*'s [36] smart ants still inherits the weakness of ABC system in terms of its limitation to symmetric network.

2.1.2.2 Co-operative Asymmetric Forward routing

Heusse *et. al* [37] advanced a cooperative asymmetric forward (CAF) routing for packet-switched networks. The most salient property of CAF is that it can be implemented in networks with asymmetric path costs meanwhile it only uses forward ants. CAF achieves this property by updating the entry in a pheromone routing table using the cost in the reverse direction of an ant traveling recorded by a data packet. So each node of CAF should maintain another table storing the reversed distance from its

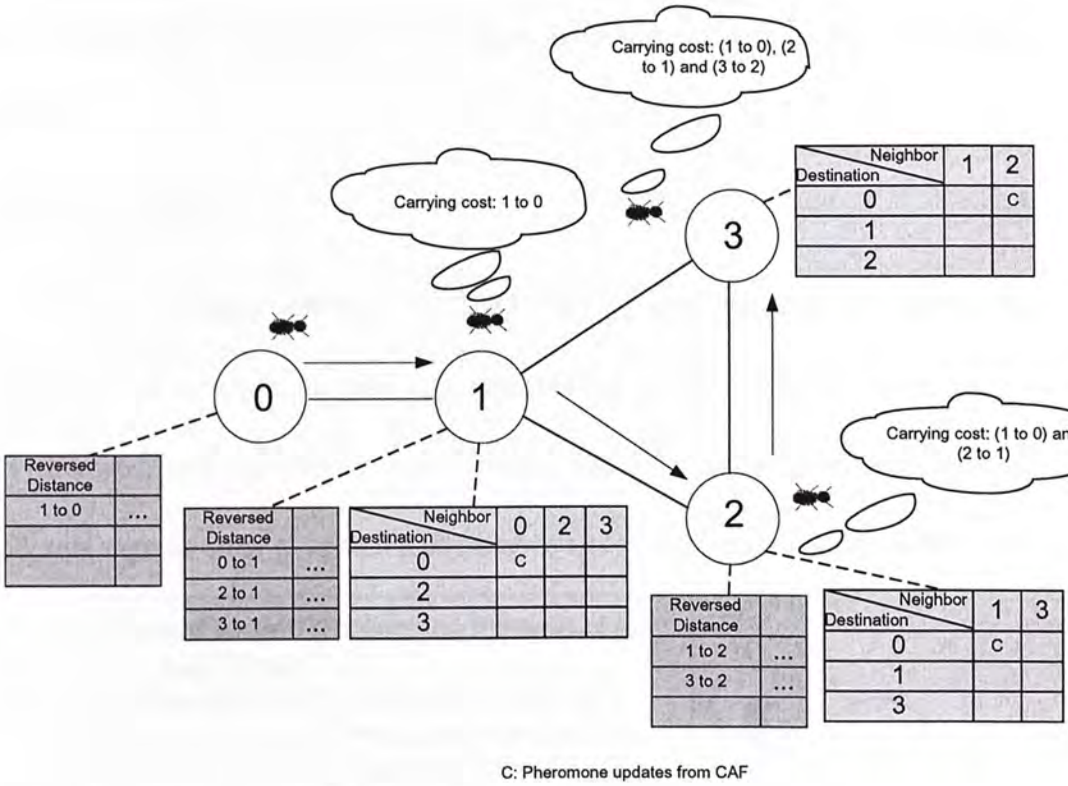

Fig 2.2. Pheromone Update Policy of CAF

neighbors (See fig 2.2). For example, there is an ant moving from node 0 to 3 through the intermediate nodes 1 and 2. When this ant leaves node 1 to node 0, it takes the distance **D1** of node 1 to 0 from the reversed distance table. In the event of its arriving at node 1, it uses this distance **D1** to update the corresponding entry in the pheromone table, i.e. the entry with neighbor 0 and destination 1 in the pheromone table of node 1. After that, it takes the distance **D2** of node 2 to 1, and uses the distance **(D1+D2)** to update the corresponding entry and so on. Consequently, CAF does not have the limitation of only applicable in the packet-switched networks with symmetric path costs. However, since CAF relies on the data packets to collect the network information from the opposite direction, it may not be appropriate when there is no frequent data traffic emerging from the opposite direction. And to make things worse, in the initial stage, it is very undesirable because the route of data packets was totally random.

2.1.2.3 AntNet

Di Caro and Dorigo's AntNet [2][22][23][38][39] was devised for routing in packet-switched networks. In their approach, the ant simply stores the record of the route followed, and the relevant age statistics, and after reaching its destination it retraces its steps in order to updating pheromone tables at the intermediate nodes.

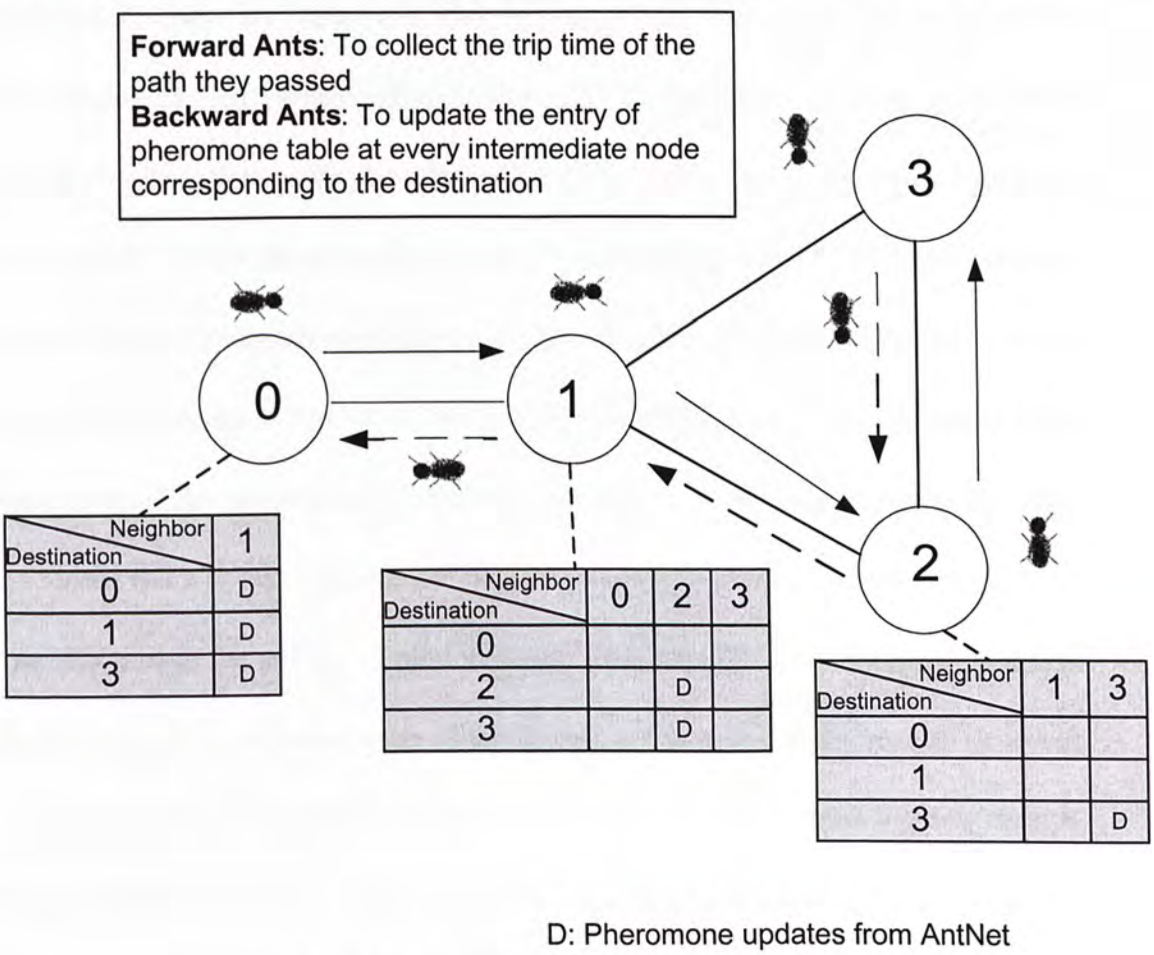

Fig 2.3: Pheromone Update Rule of AntNet.

In AntNet, routing is achieved by launching forward ants at regular intervals from a source node to a destination node that is chosen based on the data traffic pattern (in terms of the number of packets traveling from source to destination). The forward ant selects the next hop N_i using a random scheme that take into consideration of both (1) the probability P_{id} of choosing N_i, and (2) a heuristic correction factor l_{ni}. While l_{ni} is based on the queue length at N_i, P_{id} is a selection probability that can be viewed as a pheromone concentration that can be reinforced by other ants. The forward ants' job is to collect statistics such as the local data traffic condition on each intermediate node and the trip time to them on the path they have passed. After forward ants arrives the destination, they are destroyed and backward ants that store the exact same information as their corresponding forward ants are generated. These backward ants follow the same path as the their forward ants but in the reverse direction. Backward ants update (1) the probabilistic routing (or pheromone) table at each intermediate node N_i, and (2) the estimated trip time from source to N_i (gathered by the forward ant). The goodness of trip times recorded by forward ants and a squash function are used to determine the amount of reinforcement to the selection probability. Employing goodness and a squash function can be seen as some "coding" schemes that code the raw trip time to the amount of reinforcement: Goodness [2] is a relative measure that is determined by the comparison of the current trip time to (1) the current statistical estimates of the best trip time and (2) the confidence interval of the best trip time. A squash function is a non-linear function that is more sensitive in rewarding solutions with higher goodness values while having a lower tendency of reinforcing solutions with lower values of goodness. Fig 2.3 shows the pheromone update process and policy of AntNet.

By using both forward ants and backward ants, AntNet is not restricted to routing application in networks with symmetric costs only. The cost of each pair between source and destination is firstly collected by forward ants and is taken back by backward ants to update the pheromone table. In the example presented in Fig 2.3, the backward ant takes the cost from source (node 0) to destination (node 4) back to node 0 to update entries corresponding to the destination. However, using both forward and backward ants generally doubles the routing overhead and may cause late delay of updating pheromone tables.

Di Caro and Dorigo [2][22][23][38][39] use the throughput and average delay as performance measures. They conduct experiments on different topologies (NSFNET, NTTnet) using different traffic patterns such as temporal and spatial conditions [2] [38][39] and uniformly random distribution [22][23]. The experimental results of AntNet are better than that of Distance Vector, Link State, Bellmand –Ford, Shortest Path First and Q-routing.

In[40], a variant of AntNet was devised. In this variant of AntNet, forward ants travel from a source to a destination in a higher priority queues than data packets, and collect more statistics than original AntNet such as size of queuing data, links' bandwidth and delay. Backward ants update local traffic statistics, determine and deposit the amount of probability to reinforce. Since backward ants determine the amount of reinforcement using more statistics, the routing information is possibly more accurate and up-to-date. Experimental results from [40] demonstrate that the performance of this approach is comparatively better than the original AntNet.

Baran and Sosa[41] advanced another extension of Antnet. Their approach have six distinguishing features from AntNet: (1) intelligent initialization of routing table, (2) intelligent pheromone updates after link or node failures and recovery, (3) the use of noise to mitigate stagnation, (4) both deterministic and probabilistic selection of next nodes, (5) restricting the number of ants inside a network, and (6) self-destruction of Ants. Feature (1) generally reduces the exploration workload of ants to achieve the same effect in the initial stage. In the original version of AntNet, entries in a routing table consist of a uniform distribution of probabilities that do not consider the topology of the network. Taking the apriori knowledge of network topology into consideration, Baran and Sosa's ants are designed to select neighboring nodes with a higher initial probability. While original AntNet do not consider situations of network resource failures or recovery, i.e. link failure and recovery, feature (2) uses the following scheme to cope with these situations. In the event of link failures, the corresponding probability P_{ij} of the failed link *(i,j)* is set to zero and P_{ij} is distributed evenly among the remaining neighboring outgoing links. When the failed link recovers, the P_{ij} will be a commitment between P_{ij} at the failure time and the initialization time. This improvement has the advantage of retaining and reusing some of the knowledge of the nodes (such as network traffic) before the failure occurs. Feature (3) is similar to Schoonderwoerd *et al*'s [20][21][30][31][32][33] noise and Devika's uniform ants[36] where some ants select paths uniformly among neighbors without taking the pheromone table into consideration to reduce stagnancy and to improve the chance of finding a new optimal path . Feature (4) adopts a combination of a deterministic approach and a probabilistic approach, each accounting for half chance, for selecting the next hop. Feature (5) prescribes that the number of ants inside a network should not exceed four times the number of links. It is unclear whether placing such an upper bound was appropriate since no mathematical proof or empirical result is supplied. Although restricting the number of ants may reduce routing overhead and possible congestion, it also brought the side effect that lowers the frequency of launching ants. And in return, it may lead to possible reduction in the adaptiveness of the routing algorithm. Feature (6) suggests that forward ants would destroy themselves if they are going to fall into an endless loop and backward ants would self-destroyed if they could not return to the source. This feature may be trivial because in the implementation, each ant has its life. Even the algorithm had not state this feature explicitly, the implementation have actually already included this feature.

2.2 ACO Routing and Traditional Routing Algorithms

This section discusses main differences between ACO routing and traditional shortest path routing algorithms/protocols such as the Bellman-Ford algorithm[26] (another name is Distance Vector algorithm) on which RIP[27] is based and Dijkstra algorithm[28] that is used by OSPF[29] to compute the least-cost path. We do not differentiate BF algorithm with RIP, Dijkstra algorithm with OSPF though RIP and OSPF are greatly extended by a lot of mechanisms integrated into BF algorithm and Dijkastra algorithm respectively to make them applicable in the real networks.

The first main difference lies in how to construct, update and maitain the routing table. In RIP[27], each node N_i depends on the routing information provided by all its neighbor nodes. Furthermore, the neighboring nodes of N_i in turn depend on the routing information of their neighboring nodes that in turn depend on their own neighboring nodes. Each node uses the information offered by its neighbors for computing the shortest path to destinations by distributed asynchronous Bellman-Ford algorithm. While in OSPF[29], each node maintains a complete topological map (that is, a directed graph) of the entire network. Then each node locally runs Dijkastra's shortest path algorithm to determine a shortest-path tree to all networks with itself as the root node. The node's routing table is then determined by this shortest-path tree. In ACO routing, the routing table is built by ants using different schemes[2][20][21][22][23][24][36][37][40] (one may refer to section 2.1 for details). Although detailed algorithms differ from one to one, ACO routing algorithms have the following similarities. In ACO routing algorithms, the paths from a source to a destination are explored independently and in parallel. Routing information does not exchange among neighbors. Instead of that, ants take the responsibility for disseminating, exchanging and updating routing information.

Because of the "flooding" policy used by RIP and OSPF for maintaining the routing table, the routing overheads of RIP and OSPF are much bigger than that of ACO algorithms. Routing in RIP involves the transmission of routing tables of each node N_i to each of its neighbors regularly. For a large network N, the size of a routing table of N_i that encapsulates a list of least cost vectors to all other nodes in N is large. Since each N_i needs to transmit its routing table to all its neighbors, the routing overhead could be very large. In OSPF, exchange of routing information is achieved by making each node transmit a link-state-packet (LSP) to all other nodes in a network N through a *broadcasting* processing regularly. Though an LSP that encapsulates costs (the link lengths) to all the neighbors of its origin node is usually smaller than a routing table, the broadcasting process still entails a large mount of network resources. Since an LSP from a node can be disseminated via different paths to other nodes, multiple identical copies of the same LSP may be transmitted to the same node. On the other hand, transmitting ants rather than flooding the routing tables or broadcasting the LSPs accomplishes exchanging of routing information in ACO algorithms. Although the size of ants varies in different approaches, it is still very small that is in order of 6 bytes ([36], p.2).

The second main difference between ACO routing with RIP and OSPF is that RIP only use the single, shortest path between each source-destination (SD for short) pair and OSPF only allows multi-path routing among the paths with same cost, while ACO routing can be implemented as a multi-path routing among the paths that are necessarily with same cost using the probabilistic pheromone routing table. Only using the single path between each SD pair, it has several drawbacks. The first one is waste of network resource when the input load is high. It seems straightforward there are usually more than one path between each SD pair. So under the high input load condition, only using the single path while leaving other paths idle causes a great deal waste of network resource in terms of link capacity, routers memory and processing capability. Thus, using multi-path routing can utilize the network resource more efficient and increase the bandwidth between each SD pair intrinsically. Besides, in ACO routing, the pheromone value in each entry can denote the relative goodness of its neighbors[2][36]. Secondly, in the event of high input traffic, the single paths used between SD pairs may become congested soon. Consequently, a lot of packets would be dropped if the network queue was short, or the waiting time in the network queue would be extremely long if queues were long. On the other hand, in the paradigm of multi-path routing, when congestion happens, data traffic can be routed to other paths thus reduces the packet loss or waiting time in the queue.

To summarize, ACO routing algorithms distinguish themselves from traditional routing algorithms generally in terms of two aspects: (1) Routing overhead of ACO is much less than that of RIP and OSPF and thus ACO routing algorithms can save a lot of network resources (Detailed analysis needs to be done to substantiate this claim). (2) Multi-path routing of ACO algorithms is more resource efficient than single-path routing of traditional routing algorithms when the input loads are high and not balanced.

2.3 ACO Routing and Optimal Routing Algorithms

Optimal routing is a class of routing algorithms that is based on viewing data traffics as flows from sources to destinations and formulated to a multicommodity optimization problem. A lot of literatures have been published to address both the theory and applications of optimal routing ([42]-- [51]). Here, we will not go through these works one by one. We just have a look at the common framework of optimal routing. The description of optimal routing generally follows the book[52].

A communication network is often modeled as a directed graph G = (V, E), where V is the set of nodes in the network and E is the set of links in G which represents the communication channels. In this model, a group of notations are defined as follows:

W: The set of SD (Source – Destination) pairs requesting communication;

w: A generic of SD pair (i, j) in W;

F_{ij} : Traffic arrival rate (flow) on link (i, j) or on the pair w = (i, j);

r_w : The arrival rate (traffic demand) measured in data units/sec, for the SD pair w;

P_w : For a SD pair w, this is the set of all paths connecting the origin node to the destination node;

p: A generic path in P_w;

x_p : The flow rate on the path p.

The objective of optimal routing is to minimize all SD pair's cost by reasonably distribute F_{ij} to each pair w. So the cost function of optimal routing can be represented as

$$\sum_{(i,j)} D_{ij}(F_{ij}), \qquad (2.1),$$

where each D_{ij} is a function of each flow F_{ij}. The total flow F_{ij} of link (i, j) is the sum of all path flows traversing the link (i, j)

$$F_{ij} = \sum_{\substack{\text{all paths p} \\ \text{containing (i,j)}}} x_p$$

It is obvious that the collection of all path flows $\{x_p \mid w \in W, p \in P_w\}$ should satisfy the following constraints

$$\sum_{p \in P_w} x_p = r_{w,} \text{ for all } w \in W$$
$$x_p \geq 0, \text{ for all } p \in P_w, w \in W$$

By the above formulation, the optimal routing problem becomes the problem of finding the set of path flows $\{x_p\}$ that minimize cost function (2.1) subject to the constraints above. Mathematically, it can be represented as follows:

$$\min \sum_{(i,j)} D_{ij}[\sum_{\substack{\text{all paths p} \\ \text{containing (i,j)}}} x_p]$$
$$\text{subject to } \sum_{p \in P_w} x_p = r_{w,} \text{ for all } w \in W$$
$$x_p \geq 0, \text{ for all } p \in P_w, w \in W$$

The formulation or choice of cost function D_{ij} for optimal routing directly affects the output of the routing algorithm and determines the type of optimization problem. Literatures on optimal routing have considered a number of cost functions, such as the M/M/1 approximation $D_{ij}(F_{ij}) = \frac{F_{ij}}{C_{ij} - F_{ij}} + d_{ij}F_{ij}$. The type of cost functions can generally be categorized into three groups: (1). Linear functions: Make the optimal routing problem be a linear programming problem[53]. (2) Convex, smooth functions: Make the optimal routing problem be a convex programming problem[43][44]. (3) Convex, non-smooth functions: Make the optimal routing problem be a convex programming problem[46]. Group (1) can be solved using some standard linear programming methods, such as simplex method or interior point method[54]. Group (2) can be solved by feasible directions method or projection method[52][55]. In terms of group 3, the non-smooth convex objective functions can be approximated by a series of smooth convex function whose limit is that non-smooth function[46].

Though optimal routing has achieved "mathematical beauty" in both its theoretical formulation and analysis, it suffers some limitations that prevent it from real applications.

The first one is optimal routing's complexity. For example, when the objective function of an optimal routing algorithm (that depends on the problem it is solving) is a convex, smooth function, the projection method is used to solve it. However, the projection method always involves the second derivative computation and an inverse of a matrix whose size is the number of paths in the network. These computation themselves are already very complicated and to make things worse, the computation complexity increases rapidly as the network paths grow. In the event of time complexity, Wei *et.al* [48][47] has derived the time complexity $O(\varepsilon^{-2} h_{\min} N_{\max})$ for one particular projection methods, the *Gradient Projection Method* of the Goldstein-Levitin-Poljak type formulated by Bertsekas, where ε is a small positive constant, $h_{\min}$ is the diameter of the network and $N_{\max}$ is the number of paths sharing the maximally shared link. On the other hand, ACO routing algorithms do not entail any derivative computation and inverse of a matrix. What they need is some only some simple algebra. Though the detailed analysis of their computation complexity and time complexity of ACO algorithms have not presented (as far as the author knows, and this issue is also deemed by the author as a future, interesting work), intuitively, the complexity of ACO routing algorithms is much lesser than that of optimal routing.

Optimal routing's assumption on the stationary (static) traffic input is its second limitation. It seems obvious this assumption cannot be satisfied at today's communication network, i.e. Internet. Today's communication network is characterized by extremely dynamic data traffic in or out the network systems. In contrast, ACO routing algorithms do not have the stationary traffic input assumption.

The other limitations of optimal routing are centralization requirements [53], slow convergence rate [43]. However, optimal routing has its own merits because it may give people some guidelines about the ideal, optimal performance one can get. Table 2.2 lists the main difference between ACO routing and optimal routing.

Table 2.2: The Difference between ACO Routing and Optimal Routing.

	ACO Routing Approaches	**Optimal Routing**
Complexity	Low	Very High
Capability to Network Change	Adaptive	Static
Type of Methods	Heuristic method	Mathematical Programming
Theoretical Analysis	A Little Work	Much Work

Chapter 3

Reactive Ant Colony Optimization

This chapter describes the Reactive Ant Colony Optimization (RACO) routing approach. Section 3.1 establishes the problem model. Section 3.2 gives the algorithm description in great details and meanwhile presents the reasons behind each of its major design ideas.

3.1 Problem Model

This approach focuses on irregular topology packet-switched communication networks with an IP-based network layer, a simple transport layer and application layer (in the ISO-OSI terminology). Internet serves a very good example as this kind of communication network. This kind of communication networks is often represented by a directed graph G= (V, E), where V is the set of nodes in the network and E is the set of links in G that represents the communication channels.

This approach does not make any distinctions between hosts and routers. So nodes represent: (1) the network routers where route decisions are made and data packets are forwarded; (2) hosts where data sessions generate or end.

Links in the communication network can be used for bidirectional communication. Each bidirectional link should be considered as two unidirectional links. The unidirectional link l_{ij} from node i to node j represents the directly connected communication channel from node i to node j. Link l_{ji} means the unidirectional link with the inverse direction of the link l_{ij}. These links are characterized by a bandwidth (Mb/sec), a propagation delay (sec) and a queue size (number of packets). Each unidirectional link holds a queue that is a data buffer where incoming packets are stored.

In this approach, link cost is used as the routing metrics. Link cost is determined by link bandwidth, link propagation delay and link queue size. In this approach, the link cost is represented by the amount of time for a packet traversing the link that is called packet delay for short. It is reasonable because the packet delay traversing a link from a node equals to the sum of these three components: (1) the queueing delay that is packets' waiting time in the queue; (2) the propagation delay that is proportional to the physical distance between the source and destination; (3) the transmission delay that equals to packet size/ link bandwidth; (4) nodal processing delay that is the time required to examine the packet's header and determine how to switch the packet to its outgoing links. The asymmetric network means that the costs of two unidirectional links of one bidirectional link are different, that is, the link cost of l_{ij} does not equal to the cost of l_{ji}.

All the traveling packets are classified into two groups: data packets and Ants. All these packets have the same priority, so they are queued and served on the basis of a first-in-first-out (FIFO) policy.

In the case of a packet coming into a node, the node's routing component assigns an outgoing link for the packet going towards its destination node using the information stored in the local routing table. If the link queue is empty, the packet is transferred immediately. The time a packet cost to move across a link is determined by the packet size, the link bandwidth and the link propagation delay. If the link queue is neither empty nor full, this packet will wait in the queue until it gets transmission. If on a packet's arrival, the link queue has no enough buffer space to hold it, the packet is discarded.

This approach uses three parameters to adjust the behavior of RACO routing algorithm. These three parameters are *traffic load, average end-to-end delay* and *throughput*. Traffic load is defined as the ratio of the mount of traffic traversing on the link in a particular interval and the product of this link's bandwidth with this interval. Let $load_{ij}$ be the traffic load of the unidirectional link l_{ij}; B_{ij} be the bandwidth of l_{ij}; Δt be the time interval; $traffic_{ij}$ denote the amount of traffic traversing the l_{ij} in Δt. Thus, $load_{ij}$ can be denoted as follows:

$$load_{ij} = \frac{traffic_{ij}}{B_{ij}\Delta t} \quad \textbf{(3.1)}$$

Many literatures have different definitions of throughput. Here, we just use the normalized version of throughput introduced in[52]. Throughput is defined as the ratio of the difference of offered load and dropped load over offered load in a period of time.

$$throughput = \frac{offered\ load - dropped\ load}{offered\ load} \quad (3.2)$$

The value of throughput is between 0 and 1, inclusively. Average end-to-end delay is the average time of all packets moving form sources to destinations.

Fig 3.1 shows how to compute these three parameters: traffic load, average end-to-end delay and throughput. In Fig 3.1, suppose (1) there are two data flows: data flow 1 from node 3 to 0 and data flow 2 from node 2 to node 0; (2) their data rates are 1Mb/s; (3) each packet's size of both data flows is 1000 bit; (4) in the duration of two seconds, number of dropped pack for data flow 1 and data flow 2 is

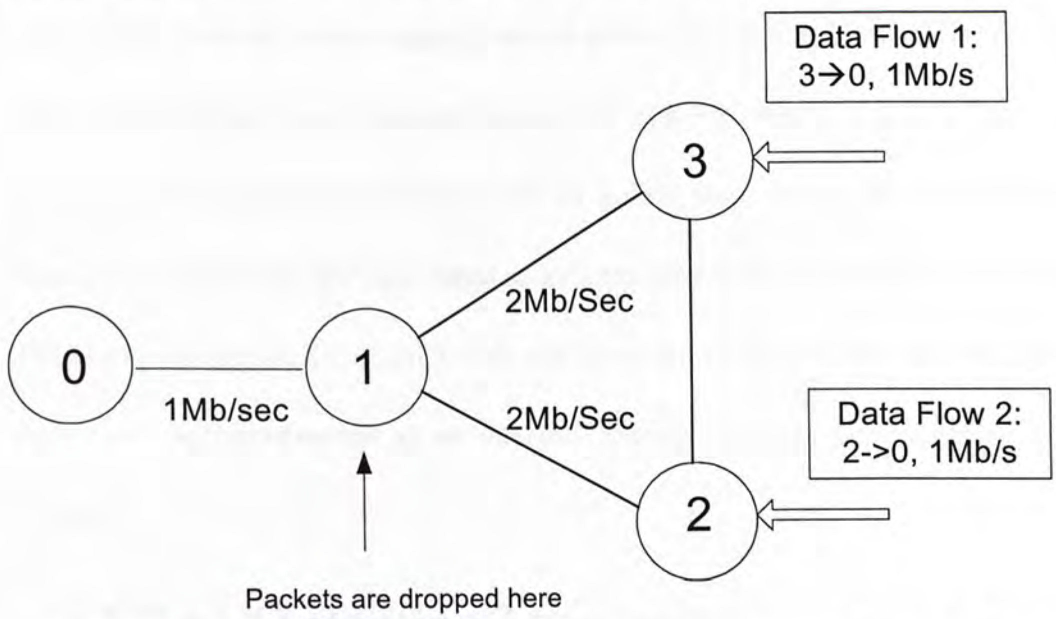

Fig 3.1: Computation of traffic load, throughput, average end-to-end delay.

1000 and 1100, respectively; (5) The delay of each packet for data flow 1 and data flow 2 is 2 sec and 3 sec, respectively. Then,

$$load_{10} = \frac{1900*10^3}{1Mb/\sec*2} = 0.95\,,\ \ throughput = \frac{4000-2100}{4000} = 0.475\,,$$

$$\text{average end-to-end delay} = \frac{2*1000+3*900}{1900} = 2.47\,.$$

Average end-to-end delay and throughput are two key parameters to evaluate the performance of a routing algorithm[52]. Citing Bertsekas and Galleger's book[52], page 367: "*In conclusion, the effect of good routing is to increase throughput for the same value of average delay per packet under high offered load conditions and to decrease average delay per packet under low and moderate offered load conditions.*"

The approach presented in this paper aims to improve both the throughput and average end-to-end delay compared with other ACO routing approaches.

We end this section with some remarks about the RACO routing algorithm itself. First, situations causing a temporary or long last change of network topology (link or node failure, recovery, or emergence) are not taken into account. Second, We do not specify this approach is an inter-AS routing or intra-AS routing approach. Because inter-AS routing approaches focus more on policy than performance[72], RACO approach should be mainly considered as an intra-AS routing approach. However, as RACO routing approach is scalable (this will be shown in the simulations results part), it also can be implemented as an inter-AS routing approach with some policies included.

3.2 RACO Routing Approach

This section introduces RACO approach in a great detail. RACO routing approach is inspired by previous work of AntNet[2][22][23][38][39] and Devika's approach[36]. However, RACO eliminates the limitation of using backward ants used in AntNet that may cause late updating pheromone values and overcomes the Devika's approach's shortcoming that it is only applicable in symmetric networks.

To give readers a whole picture of RACO routing approach, it can be summarized as follows:

- At regular intervals, from each network node RRAs are asynchronously launched towards randomly chosen destination nodes.
- RRAs act concurrently and independently, and communicate in an indirect

way, by the information they read and write into nodes locally.

- RRAs collectively search for shortest paths connecting each pair of source and destination.
- While traveling, each RRA collects the network information about the link delay, detects the congestion status and node identifies of the path it has passed.
- Each agent moves step by step towards its destination node. Upon arriving a node, this ant will (1) check if network congestion happens, RCCA will be triggered to alleviate it; (2) write the information it has collected into the local statistics of the node; (3) update the entry of the routing table in this node corresponding to its source node by some updating rules; (4) Destroy itself if the node is its destination node or continue its traveling toward its destination node.

In the following subsections, the above scheme is explained, all the related components are discussed, some properties are derived and a more detailed description of this approach is given.

3.2.1. Node and Probabilistic Routing Table

Nodes represent routers where route decisions are made or hosts where data sessions generate or end. They mainly assume the following tasks:

1. To generate RRA periodically, asynchronously for exploring the network and constructing the routing tables;
2. To generate RCCA according to the network status to alleviate the network

congestion;

3. To receive both ants and data packets and forward them according to the routing table;

4. To maintain and update the routing table, local and global cost statistics regularly.

Fig 3.2 illustrates the structure of a node R_k that has **L** neighbors and the network has **N+1** nodes. There are three basic routing-related components in the node: routing table, local cost statistics and global cost statistics.

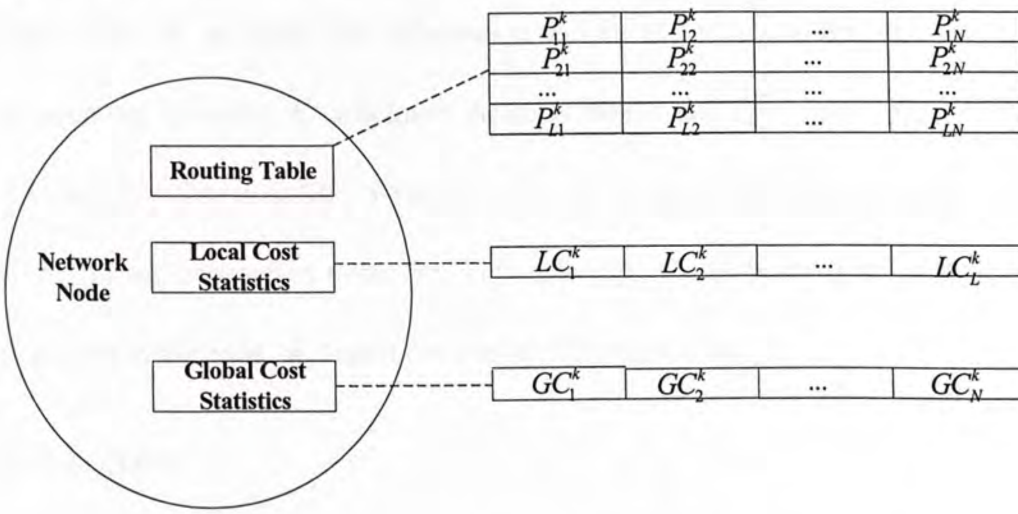

Fig 3.2: Node Structure for the Case that the node R_k has L neighbors and a network with N+1 nodes.

(1) *Routing Table*: As node R_k has **L** neighbors and the network has **N+1** nodes, the routing table in the node R_k looks like a **L** by **N** matrix. The routing table is also called as the pheromone table in this RACO routing approach. The value of P^k_{ij} in each entry actually denotes the ratio of the pheromone amount on the neighbor link connecting to node R_i over the pheromone amount of its all neighbor links. P^k_{ij} also denotes the probability of an ant or a data packet choosing the neighbor R_i towards the destination node R_j. It is obvious that the summation of all row entries

in the routing table equals to one. That is:

$$\sum_{n \in N_k} P_{nd}^k = 1 \quad ,$$

where $d \in [1, N+1]$, $d \neq k$, $N_k = \{neighbors(k)\}$. The probabilistic routing tables are a mechanism for exploring alternate paths in the network and thus RACO can be implemented as a multi-path routing algorithm. The probability values are uniformly distributed among a node's neighbors at the beginning.

(2) *Local Cost Statistics.* LC_i^k ($i \in N_k$) in node R_k stores the newest delay from its neighbor node R_i to itself. This information will be taken back to the node R_i by an ant such that the node R_i will know the delay from node R_i to node R_k.

(3) *Global Cost Statistics.* GC_j^k ($j \in [1, N+1]$, $j \neq k$,) stores the smallest delay from node R_k to the destination node R_j. This information will be used to update the routing table in the node R_k upon the ants' arrivals from node R_j.

3.2.2. Ants

Two different types of ants used in RACO approach are discussed: one is RRA (Reactive Routing Ants) whose job is to collect the network information, to find the shortest paths between each pair of source and destination, and to construct the routing tables; the other is RCCA (Reactive Congestion Control Ants) that is responsible for alleviating the congestion. The underlying principle of devising these two types ants follows the instructions on [52] (p 454): *for low input OD pairs, each OD pair tends to use only one path for routing (the fastest in terms of packet transmission time), and as traffic input increases, additional paths are used to avoid overloading the fastest path.* So roughly speaking, RRA is to find the shortest path and RCCA is to find the additional paths as input traffic increases.

3.2.2.1. RRA (Reactive Routing Ants)

RRA is responsible for exploring the network and constructing the routing table. Its behavior to eliminate the use of backward ants is partially inspired by[37]. Periodically, each node (router) R_s in the network generates a RRA to another random chosen node (router) R_d as the RRA's destination. The RRA is of the form (R_s, R_d, C_e, C_c). C_e is the RRA's experienced cost (delay) when it moves from one node to another. C_c is the RRA' carrying cost (delay) when it migrates in the network. The computation of C_e and C_c will be explained in the following paragraphs.

When a RRA is about to leave the node R_i, it first chooses its next node(suppose it is R_j). If R_j has been visited, the RRA would choose its next node again. If all the neighbors of R_i have been visited, the RRA destroys itself immediately. If R_j has not been visited, then it reads the newest cost LC_j^i (the cost from R_j to R_i) from the local statistics of the node R_i. The carrying cost C_c equals to the summation of original carrying cost $C_c^{'}$ and LC_j^i: $C_c = C_c^{'} + LC_j^i$. Upon arriving the node R_j, the ant writes its experienced delay (C_e that is from node R_i to node R_j) to the local statistics LC_i^j of the node R_j and update the routing table's entries corresponding to the its source R_s using the following updating rules:

$$\begin{cases} P_{ks}^j(t+1) = \dfrac{P_{ks}^j(t) + \Delta p}{1+\Delta p}, \text{ when } k = i \\ P_{ks}^j(t+1) = \dfrac{P_{ks}^j(t)}{1+\Delta p}, \text{ when } k \neq i,\ k \in N_j \end{cases} \qquad (3.3),$$

where

$$\Delta p = f(GC_s^j, C_c) = \begin{cases} \dfrac{GC_s^j}{C_c}, & \text{if } GC_s^j \geq C_c \\ 0.1\dfrac{GC_s^j}{C_c}, & \text{if } GC_s^j < C_c \end{cases} \quad (3.4).$$

In equation (3.4), GC_s^j is the least cost from node R_j to this RRA's source node R_s; Δp is the pheromone update rate that plays an essential role in the convergence behavior of RRA (This will be shown in Chapter 5.).

After updating the routing table, the RRA also updates the global cost statistics presented as follows: If the RRA's carrying cost C_c is not bigger than GC_s^j, then

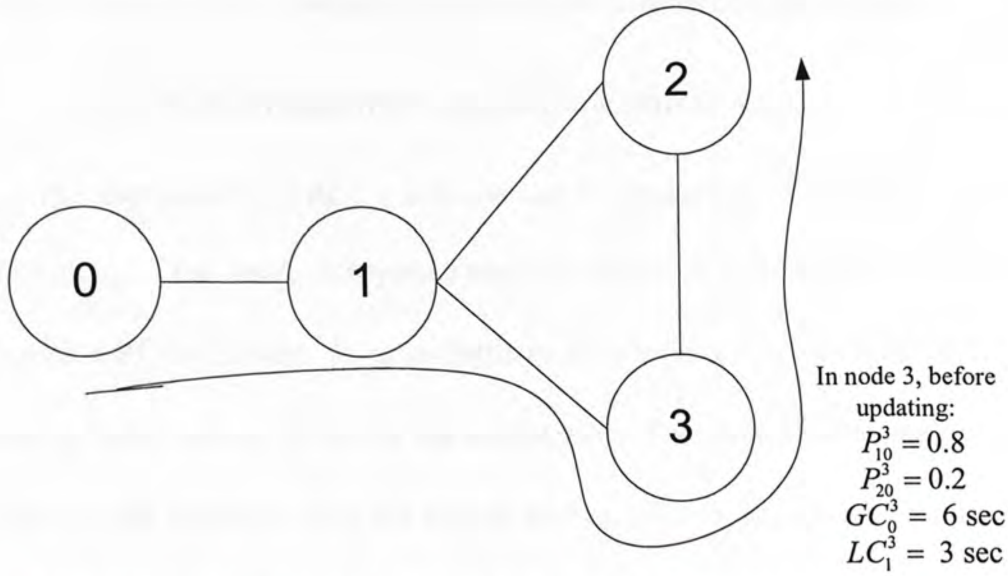

Fig 3.3: Illustration of RRA's behavior. A RRA ant moves from node 0 to node 2 thorough intermediate nodes 1 and 3. It will update the entries corresponding to destination node 0 of routing table in node 1, node 3 and node 2.

$GC_s^j = C_c$; otherwise, GC_s^j remains no change.

A simple example is helpful to understand the mechanism of RRA. Suppose in Fig 3.3, a RRA ant with source node 0 and destination 2 chooses the path 0 → 1→ 3→ 2. When it leaves node 1 for node 3, it carries $C_c = LC_1^0 + LC_3^1$ to node 3, where

LC_i^k means the newest delay from node **i** to the node **k**. When the ant arrives the node 3, it tells the node 3 its experienced delay C_e traversing the link l_{13} and thus LC_1^3 is updated to newest delay from node 1 to node 3. After that, it updates the routing table in node 3 according to the equations (3.3)(3.4). If C_c=5 sec that is smaller than GC_0^3, then $\Delta p = \frac{6}{5}$, $P_{10}^3 = 0.91$, $P_{20}^3 = 0.09$ and $GC_0^3 = C_c = 5$ sec; if C_c=7 sec that is bigger than GC_0^3, then by equation (3.4), $\Delta p = 0.1 * \frac{6}{7}$ =0.086, $P_{10}^3 = 0.82$, $\mathrm{P}_{20}^3 = 0.18$.

If $R_j = R_d$ that means the RRA reaches its destination, the RRA agent self-destroyed. If not, it continues its traveling towards its destination node R_d.

3.2.2.2. RCCA (Reactive Congestion Control Ants)

The responsibility of RCCA is to alleviate the congestion. When a RRA traverses the link l_{ij}, if the $load_{ij}$ is beyond a particular threshold γ, it will inform the node j to emit a RCCA to node R_i to re-distribute the pheromone value on the node R_i's routing table, namely, to update the routing table. The re-distribution only happens between two neighbors with the biggest and second –to- biggest probability value with respect to the node **d**, which is the destination node of data packets causing congestion.

The redistribution happens as follows. Suppose in node R_j, P_{jd}^i and P_{kd}^i ($k \in \{neighbors\ of\ node\ i\}$, $k \neq j$) are the biggest and second-to-biggest probability value to the destination node **d**, respectively. $P_{jd}^{i'}$ and $P_{kd}^{i'}$ are the values after redistributed.:

$$\begin{cases} P_{jd}^{i'} = \alpha P_{jd}^{i} + \beta P_{kd}^{i} & (3.5.1) \\ P_{kd}^{i'} = \alpha P_{kd}^{i} + \beta P_{jd}^{i} & (3.5.2) \end{cases} \quad (3.5)$$

where $\alpha = 1 - load_{ij} / C$, $\beta = load_{ij} / C$. The positive constant **C** is called as the coefficient of RCCA.

This kind of pheromone redistribution has the following properties.

Proposition 3.1: With the (3.5) pheromone redistribution scheme, the heavier the link load is, the more pheromone is redistributed to another link. Mathematically, if $load_{ij}^{'} > load_{ij}^{"}$, then $P_{kd}^{i'} = f(load_{ij}^{'}) \geq P_{kd}^{i"} = f(load_{ij}^{"})$.

Proof:

$$\begin{aligned} P_{kd}^{i'} - P_{kd}^{i"} &= (1 - \frac{load_{ij}^{'}}{C}) P_{kd}^{i} + \frac{load_{ij}^{'}}{C} P_{jd}^{i} \\ &\quad - (1 - \frac{load_{ij}^{"}}{C}) P_{kd}^{i} - \frac{load_{ij}^{"}}{C} P_{jd}^{i} \\ &= \frac{1}{C} (P_{jd}^{i} - P_{kd}^{i})(load_{ij}^{'} - load_{ij}^{"}) \end{aligned}$$

Because $P_{jd}^{i} \geq P_{kd}^{i}$, $load_{ij}^{'} > load_{ij}^{"}$, and **C** is positive, we have $P_{kd}^{i'} - P_{kd}^{i"} \geq 0$. **Q.E.D.**

Proposition 3.2: The constant **C** bounds the pheromone difference, that is, $P_{jd}^{i} - P_{jd}^{i'} \leq \frac{1}{C}$.

Proof:

$$\begin{aligned} P_{jd}^{i} - P_{jd}^{i'} &= P_{jd}^{i} - (\alpha P_{jd}^{i} + \beta P_{kd}^{i}) \\ &= \frac{load_{ij}}{C} (P_{jd}^{i} - P_{kd}^{i}) \qquad (3.6) \end{aligned}$$

Because $load_{ij} \leq 1$ and $P_{jd}^{i} - P_{kd}^{i} \leq 1$, equation (3.6) implies $P_{jd}^{i} - P_{jd}^{i'} \leq \frac{1}{C}$. **Q.E.D.**

Proposition 3.1 shows that the pheromone redistribution is intelligent and reactive in the sense that it takes the network status into account. The heavier the link load is, the more traffic will be arranged to another link. It is expected that this scheme results in a good performance.

The implication of Proposition 3.2 is that the amount of traffic moved from the congested link to another link bounds in terms of the positive constant C. It avoids the danger of "oscillation".

3.2.3 Detailed Description of RACO routing approach

The detailed and formal description of RACO is presented as follows:

1. At regular intervals Δt from every network node R_s, a RRA of the form (R_s, R_d, C_e, C_c) is generated towards a random chosen destination R_d.
2. Before leaving the each node R_i, the ant consults the entry in the R_i's routing table and it chooses its next node R_j with probability P^i_{jd}. Suppose the next node is R_j. Then it carries the newest cost LC^i_j (the cost from R_j to R_i) to node R_i and writes this value into C_c, where $C_c = C_c^{'} + LC^i_j$, $C_c^{'}$ is the original carrying cost ($C_c^{'}$=0, if $R_i = R_s$).
3. If a cycle is detected, that is the RRA is about to visit an already passed node, the RRA returns back to its previous node R_i and chooses among its neighbors again. If all of its neighbors have been chosen, the RRA is destroyed immediately.
4. While traveling on a link $link_{ij}$, the RRA records the delay passing this link $link_{ij}$ to C_e and detects the load on $link_{ij}$.
5. Upon arriving a node R_j, the RRA does the following five jobs: (1) If the load on $link_{ij}$ is beyond some threshold γ. A RCCA will be triggered to inform the RRA's previous node R_i to redistribute its pheromone value P_{kd}, $k \in \{\text{neighbors of node } R_i\}$ by the equation (3.6); (2) The RRA writes its experienced delay C_e into the node R_j's local statistics LC^j_i that denotes the

newest delay from node R_i to node R_j. This delay will be taken back to the node R_i by another ant to let the node R_i know the delay from R_i to R_j; (3) Update the entries in the routing table of R_j by the updating rules (3.3) and (3.4); (4) Update the global cost statistics in node R_j; (5) If $R_j = R_d$, the RRA agent self-destroyed. If not, it continues its traveling towards the destination node R_d.

Chapter 4

Simulation Results

This chapter demonstrates the simulation results of RACO and other routing algorithms for comparison. Experiment designs, such as testbed, topology and traffic pattern, are presented in section 4.1. Simulation results are demonstrated in section 4.2. Section 4.3 discusses the simulation results.

4.1 Experiment Design

NS (Network Simulator)[25] is used to implement the RACO routing approach. NS is a well-known discrete event simulator in the networking research field. It provides substantial support for simulation of network protocols or behaviors at different layers, such as CBR, Telnet and FTP in the application layer, UDP and TCP in the transport layer, unicast routing and multicast routing in the network layer.

Our testbed extends the node architecture offered by the NS to support the RACO's probabilistic routing table.

One classical shortest-path routing approach – DV[27] and two famous ACO based routing approaches – Devika's approach[36] and AntNet [2][22][23][38][39] are chosen for performance comparison. DV has already been implemented in NS. Devika's approach is only applicable in the symmetric network because of its mechanism of collecting the network information, so it cannot directly implemented into an asymmetric network. Thus, the way of acquiring the network information of

RACO has been plugged into Devika's approach while leave other components of her approach intact. There are many versions of AntNet. Among all versions, there is only a little difference. The latest version of AntNet[2] has been implemented.

4.1.1 Topology

Topology can be derived from on the basis of a real net instance or can be defined by hand to better analyze some issues of routing approach, such as scalability.

Nodes in real networks present routers or hosts, while in this testbed, it does not make any distinction between routers and hosts. In this testbed, nodes can be both routers where route decisions are made and hosts where data traffic are generated and received.

Each link is bi-directional that means packets can move between two nodes in both directions. At the initial state, the bandwidth and delay between the two directions of the same link is identical. While the simulation is running, because the traffic loads between two nodes are different, the bandwidth and delay between the two directions of the same link will not be identical. Thus, the network is not symmetric. Symmetry means the bandwidth and delay between two directions of the same link is identical. In this testbed, links are characterized by their propagation delay (sec), bandwidth (Mb/sec) and queue size (number of packets).

Each link holds two buffers in two opposite directions. Buffers represent locations where packet may be held or dropped. Buffer management, which means any particular policies used to regulate the occupancy of a particular buffer, is drop-tail (FIFO) queuing. Drop – tail means when a buffer is full, the new arrived packets to this buffer will be dropped. The buffer size is 50 packets in the simulation.

- To show the scalability of RACO, two different topologies are implemented in the testbed: one is the medium scale – NSFENT, and the other is large scale – ARPANET.
- *NSFNET.* This topology is the old USA T1 backbone – NSFNET. NSFNET is composed of fourteen nodes and 21 bi-directional links with a bandwidth of 2Mbit/s and propagation delays ranging from 4ms to 20ms. Figure 4.2 shows the topology of NSFNET.

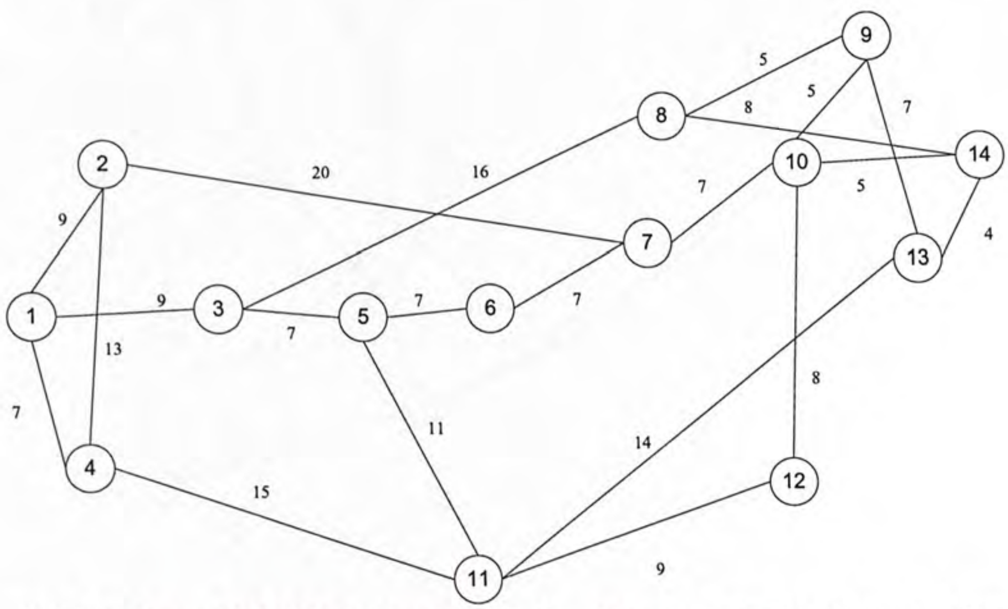

Fig 4.1. ***Topology of NSFNET.*** **Numbers in the circle are node identifiers. Each link in the graph represents a pair of directed link and the numbers near them are transmission delay in msec. The bandwidth is 1.5 Mb for every link.**

- *ARPANET.* ARPANET (Advanced Research Projects Agency Network) is the global Internet's progenitor. It is composed of forty-eight nodes and sixty-six bi-directional links with a bandwidth of 0.5Mbit/s and propagation delays of 4ms. The topology of APPANET[68] is illustrated in Fig 4.2.

4.1.2 Network Layers

This approach focuses on irregular topology packet switched, connection-less communication networks with an IP-based network layer and a simple transport layer

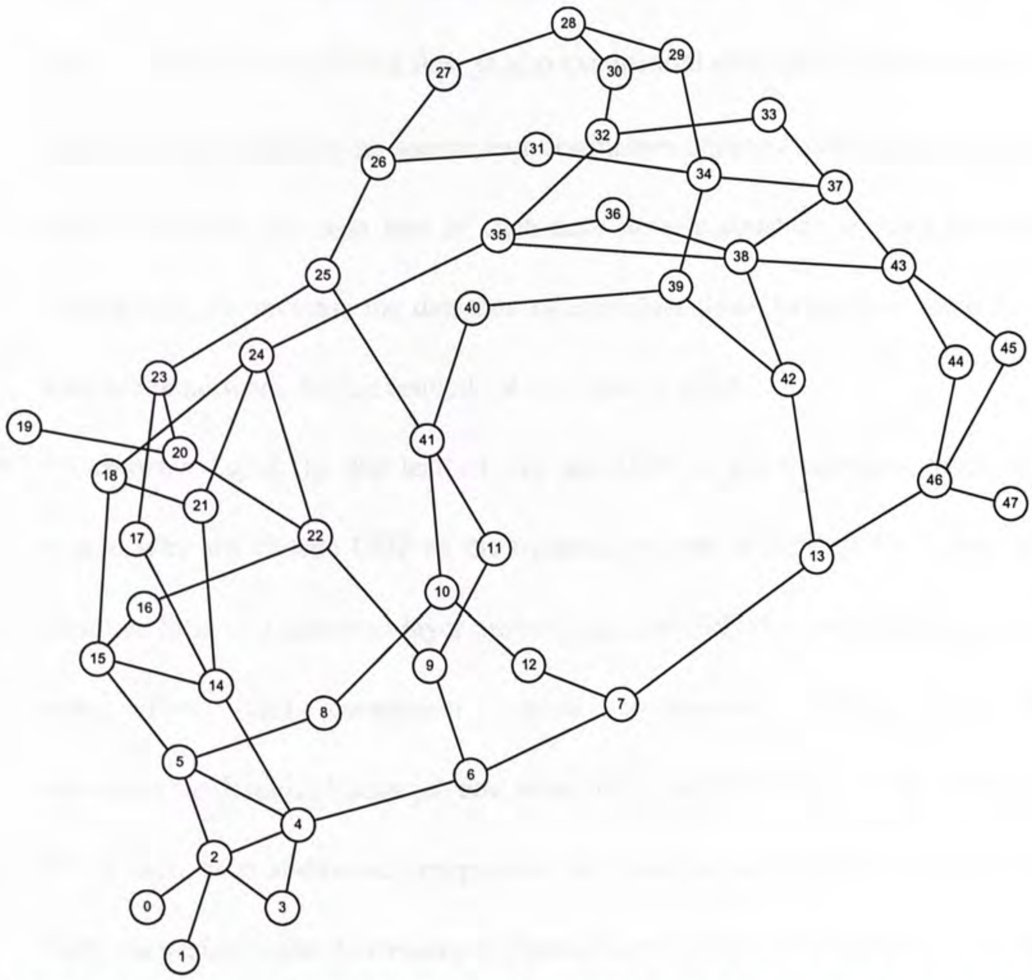

Fig 4.2. Topology of ARPANET. Numbers in the circle are node identifiers. Each link in the graph represents a pair of directed link. The transmission delay is 4 mses and the bandwidth is 0.5 Mb for every link

and application layer (in the ISO-OSI terminology).

- *Application Layer*: In application layer, this testbed implements a traffic pattern that tries to emulate the traffic in real life as "close" as possible. The traffic pattern is defined as in terms of a group of data flows between pairs of different nodes. It is characterized by the distribution of data flows' inter-arrival and duration. Each data flow is determined by its data rate and its source – destination. This simulation models the arrival of data flows as a Poisson Process in the sense that the distribution of data flows' inter-arrival is exponential distribution having rate λ. The duration of data flow is also exponential distribution having rate μ. Each data flow chooses its source and destination uniformly distributed. During each simulation, the data rate of each data flow is constant. Among different simulations, we increase the data rate of each data flows to increase the offered load to the network. In this testbed, $\lambda = 1/3$ and $u = 0.8$.

- *Transporter Layer*. In this testbed, we use UDP as the transporter layer. The reason why we choose UDP as the transporter layer is because UDP does just about as little as a transport layer protocol can do. UPD does not incorporate any error, flow, and congestion control components. Asides from the multiplexing/demultiplexing job and some little error checking, it adds nothing to IP. In fact, each additional component may have a considerable impact on the network performance. For example, Peterson and Davie [70] reported a 2 to 30% improvement in different performance measures for real Internet traffic[71] of TCP Vegas version over TCP Reno version. By using UDP as the transporter layer, we can test the routing approach separately. However, the study of interacting between RACO routing approach and different protocols in transporter layer can be a candidate for future work.
- *Network Layer*. RACO routing approach and other three routing approaches are

implemented in this layer. By increasing the data rate of each data flows during one simulation, we want to compare the performance of these three approaches.

- *Link Layer and Physical Layer*. Nothing has implemented particularly in these layers. All the related the jobs are left to NS 2[25].

4.2 Results

Experiment results reported in this section compare RACO with competing routing approaches described in section 4.1. Performance of these approaches is studied with the increasing load. Throughput and average end-to-end delay are chosen to be parameters for performance evaluation. The simulation time lasts 60 virtual seconds. Before data flows are put into the networks, all these algorithms are given 40 virtual seconds to build initial routing tables. In this simulation, the threshold for emitting RCCA is 0.7 and the coefficient of RCCA is 4. These parameters are heuristic values.

4.2.1 NSFNET

Fig 4.3 illustrates the throughput on NSFNET. It shows that when the input data rate is not fast, i.e.1.0Mb/s, the throughput of DV is better than RACO. However, with the input data rate increasing, the throughput of RACO always achieves a better performance than DV. RACO has 23.8% more throughput than DV, 14.7% more than DE and 25% more than AntNet.

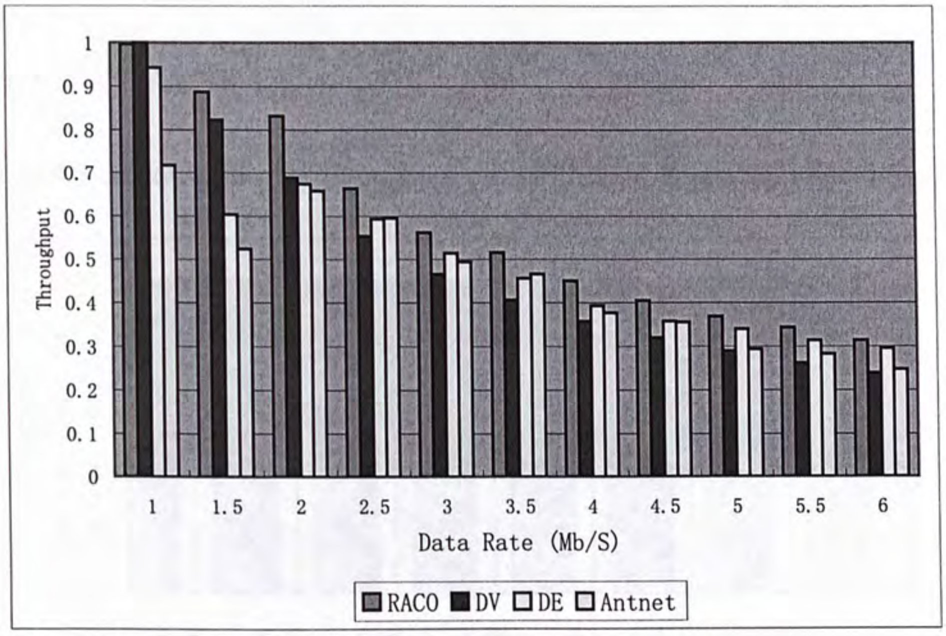

Fig 4.3. Throughput on NSFNET

Fig 4.4 shows the average end-to-end delay on the NSFNET.

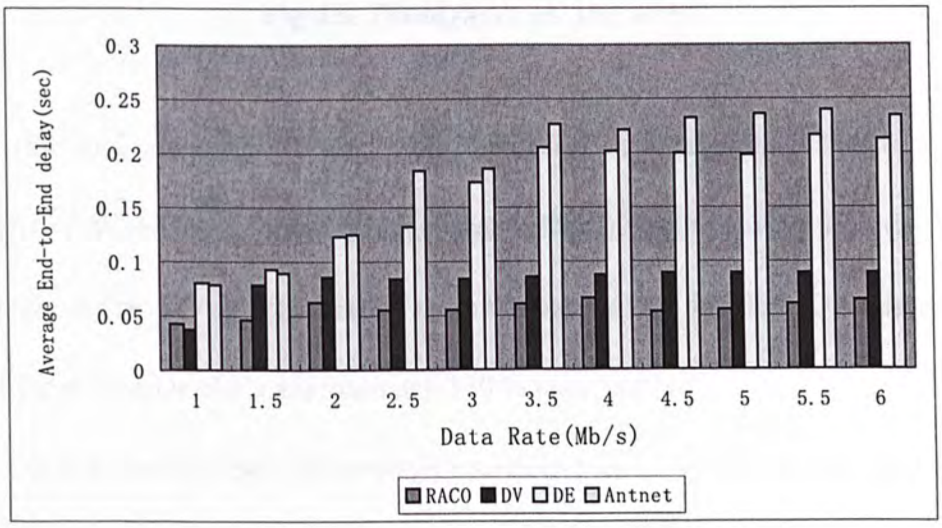

Fig 4.4. Average End-to-End Delay on NSFNET

RACO always has a better average end-to-end delay performance than DV except the case that data rate is 1.0Mb. RACO has 2.5% less delay than than DV.. Fig 4.4 also shows that in all cases, Devika's approach and AntNet have a much worse delay result than the other two no matter the traffic load is high or low.

4.2.2 ARPANET

Fig 4.5 demonstrates the throughput on ARPANET. When the offered load is not

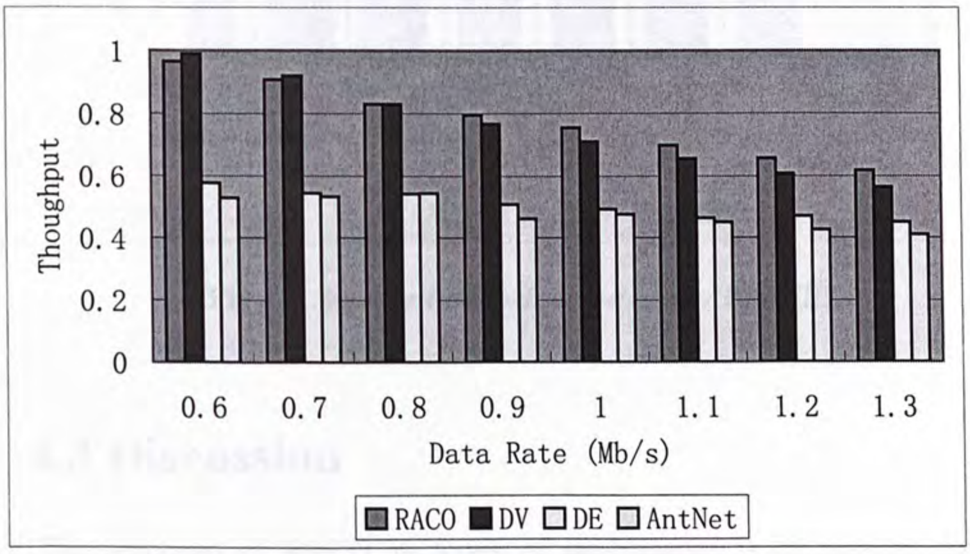

Fig 4.5. *Throughput on ARPANET.*

high (the data rate is 0.6 Mb/s, 0.7Mb/s, 0.8Mb/s). DV's throughput is better than that of RACO. When the data rate is bigger than 0.8Mb/s, RACO always achieves a better throughput than DV: it achieves 7% more throughput than DV. RACO's throughput is 57% more than Devika's approach and 160 % than AntNet.

Fig 4.6 demonstrates the average end-to-end delay on ARPANET. The average end-to-end delay of DV is more than that of RACO about 8% when input data rate is bigger than 0.8Mb/s. The average end-to-end delay of Devika's approach and AntNet are much worse than RACO and DV.

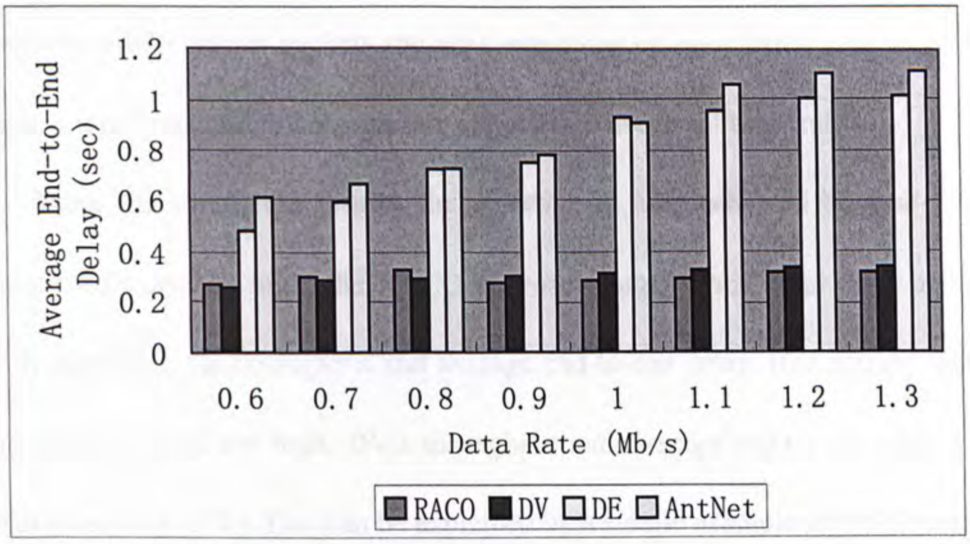

Fig 4.6. ***Average End-to-End Delay on ARPANET.***

4.3 Discussion

The objective of RACO in terms of performance is to improve the both throughput and average end-to-end delay when the offered load is high. It is coherent with statement of Bertsekas and Galleger's ([52], page 367): *"the effect of good routing is to increase throughput for the same value of average delay per packet under high offered load conditions"*. To achieve this objective, two different ants are devised. One of ant is called Reactive Routing Ants (RRA), the other one is Reactive Congestion Control Ants (RCCA). RRA is responsible for exploring the network, constructing the routing table and finding the shortest path between each source and destination. RRCA is emitted when some congestion will occur or occurs in the network for re-distributing the pheromone value from the congested path to another alternative path. The underlying principle of devising these two types of ants is that when the input traffic load is not high, each source and destination pair should use the shortest path to deliver packets and with increasing of input traffic load may result in some congestion, additional paths are expected to use to split the traffic.

From the simulation results, the objective is fully achieved because when the input traffic load is high, the RACO approach outperforms other three approaches with respect to the throughput and average end-to-end delay. It is noticed that when the traffic load is not high, DV's throughput and average end-to-end delay is better than those of RACO. This can be explained by a simple example in the Bertsekas and Gallager's book ([52], page 453-454). Consider the two-link network shown in fig 4.7, where node 1 and node 2 are only source and destination, respectively. Suppose there is an input data flow r that is to be divided into the two path flows x_1 and x_2 to minimize a cost function based on M/M/1 approximation.

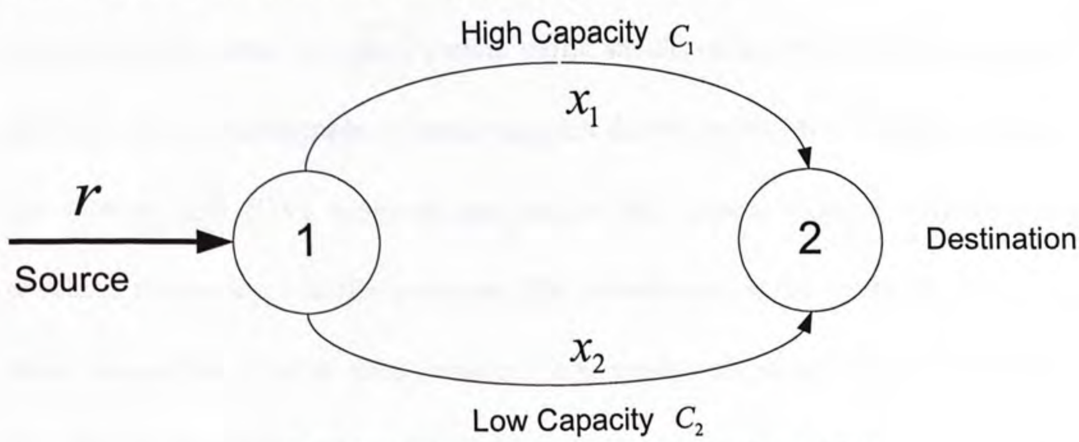

Fig 4.7. A Two-Link Network ([52], Page 453)

By using some simple calculation (For details, please refer to [52], page 453-454), it reaches the result illustrated in fig 4.8.

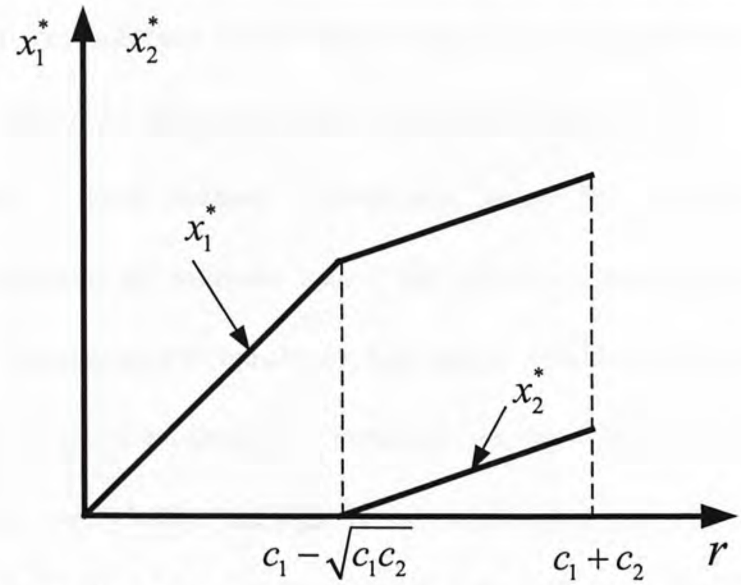

Fig 4.8. Optimal Path Flows for the Routing Example. ([52], page 454)

Fig 4.8 shows the distribution of two path flows to achieve the minimum cost (optimal value) of the routing problem. When the input traffic load is low that is, $r \le c_1 - \sqrt{c_1c_2}$, only the high-capacity path x_1^* should be used. As the input increases beyond the threshold $c_1 - \sqrt{c_1c_2}$, some traffic should be routed on the low capacity path x_2^*. From this example, it seems apparent that when the input traffic is not high, the shortest path (DV) approach can achieve the optimal solution of the routing problem. As the input traffic increases, DV cannot achieve the optimal solution any more. In contrast, RACO splits some traffic to another alternative path. Thus, RACO outperforms the DV in terms of both throughput and average end-to-end delay when the input traffic is high.

There are several reasons about why Devika's approach[36] does not have a good performance. The first reason is that the pheromone updating rules of Devika's approach may not necessary converge to optimal path, or even not converge. It remains an open problem. The second reason is its use of uniform ants. The use of uniform ants may alleviate the so-called "stagnation" problem. But it also may increase the probability of choosing some non-optimal paths. This kind of pheromone re-distribution is blind because uniform ants choose the next link uniformly distributed among all the available links. It may choose a link where the pheromone has already been stagnated (originally the best link) or a link with the least pheromone value (originally the worst link). As a matter of fact, stagnation is not a big problem when the input traffic load is not high. To only use the best path one have is always the best strategy when the input traffic is not high. This has been already illustrated in the fig 4.7 and fig 4.8. When the input traffic load is high, we may distribute the load to another alternative path. However, this distribution should not be blind. On the other hand, we should distribute some load to some paths near the best one.

From the simulation results, AntNet's [2][22][23][38][39]performance is not good. AntNet[2][22][23][38][39] involves to a complex process to encode some raw values of network information (delay) to some quantity called "goodness". Then probability for choosing next links is proportional to the links' goodness. This idea seems reasonable at first glance. However, further analysis shows that it may not be the case. Consider a simple network shown in Fig 4.9. The numbers beside the links mean the cost of each link. The probability P_{ij}^{k}, roughly representing the value of goodness, means the probability of node k chooses the next node **i** towards the destination node **j**. Suppose there is a data flow from node 1 to node 4. When the input load is not high, by the example illustrated in Fig 4.7 and Fig 4.8, any routing approach should choose the shortest path to achieve the optimal state. However, AntNet still chooses some non-optimal paths. This kind of behavior makes AntNet be far away from the optimal state. Thus, its performance will deteriorate. When the input traffic is high, it is good to distribute some traffic to other paths. However, how to distribute should follow the principle of optimal routing. From the simulation results, the distribute mechanism of AntNet is not efficient. And thus it causes large delay and consumes lots of network resource. And also because this large delay, packets cannot arrive at the nodes as soon as possible the throughput of AntNet is also very unsatisfactory.

Chapter 5

Convergence Analysis

In the RACO routing approach, RRA's job is to find the shortest path between each pair of nodes. Thus, the problem of whether the RACO routing approach using RRA to find the shortest path will finally converge to a shortest path is important to analyze. This chapter addresses this issue. Results show that RACO routing approach by using RRA to search the shortest path can converge to the optimal path under some network conditions. The mathematical analysis also shows that pheromone update rate plays a critical role in the convergence behavior.

RACO is an algorithmic variant of ACO based routing approach. Recently, the application of ACO in network routing has received a lot of attention [20][21][22][23][2][24]. However, the results of above works are mainly heuristic and experimental, the theoretical nature of ACO routing algorithms has not been thoroughly studied, such as convergence property. Thus, this work tries to analyze the convergence property of RACO. This work is also the first attempt to address the convergence analysis of ACO routing algorithms in a scenario of asymmetric network.

This chapter is organized as follows. Section 5.1 gives a short review about all the works of convergence analysis regarding ACO approaches, not limited to ACO routing algorithms. Two-Node, Two-Path model is presented in section 5.2. The equations of recursive pheromone value are derived in section 5.3. The detailed analysis is demonstrated in section 5.4 and section 5.5 considers the general models. Section 5.6 gives the conclusion to this chapter.

5.1. Related Work

Gutjahr[56][57] demonstrated an ACO variant GBAS (Graph-Based Ant System) converged to optimal solution with the probability $1-\varepsilon$, when ε can be made arbitrarily small by choosing suitable S (number of ants) and the value of the so-called evaporation factor ρ. However, the convergence probability $1-\varepsilon$ cannot be predicted by choosing S and ρ. So the convergence remains uncontrollable in the sense that even users wanted to adjust the parameters S and ρ of their systems, they could not get a guaranteed convergence result with a predicable minimum probability.

Gutjahr [58] advanced another convergence analysis for a variant of GBAS, called as the time dependent GBAS. It revealed that this particular stepwise ACO algorithm for GBAS could definitely converge to the optimal path by using some particular parameter schemes. However, it made an assumption that the pheromone update must be a Markov process that could not be applied to some ACO applications in communication network such as AntNet [2][22][23][38][39], Schoonderwoerd's ABS[20][21][30][31][32][33] and also the work presented in this thesis. In these applications, when a group of ants populates in a network to update pheromone simultaneously, it results in a non-Markov process in the sense that the current state of pheromone not only depends on the current previous state of pheromone, but also hinges on previous states of pheromone on other nodes as well.

Thomas and Dorigo [59] also proposed a convergence proof for a class of ACO algorithms, or specifically, MAX-MIN ANT system developed by Stutzle and Hoos[60][61]. For this ACO scenario, Thomas and Dorigo [59] proved two results: (1) the probability of finding the optimal solution at least once could be greater than $1-\varepsilon$ for any small constant $\varepsilon > 0$ as long as the algorithms would run a sufficiently large number of iterations; (2) After the optimal solutions were found, the pheromone values would be higher on the links belonging to the optimal solution than on any other connection after some time.

The works [56][57][58][59] discussed in previous sections only analyzed the ACO with the application to TSP. An important property of ACO with TSP application is that the pheromone updating only occurs at the end of each iteration. Thus the pheromone updating process is a Markov chain. In contrast, pheromone updating is done in parallel, independently and at any time in the scenario of ACO with network routing application. Hence, convergence analysis in this work is treated differently from the works [56][57][58][59]. Devika [36] studied one special case of ACO with network routing application. In this case, a model of static 2-path selection with one pheromone table locating in each node was analyzed. Devika [36] claimed that regular ants could converge to the shortest path. Unfortunately, the proof presented in Devika [36] was not rigorous and could be even wrong. The first flaw in its proof is that it assumes that the increasing the pheromone value twice and decreasing once must be bigger than the original value, which cannot always be true. Moreover, Devika [36] only studies the case that the cost of path is twice as the other one, and then it claims that it can be generalized to more general cases without detailed proof. Hence, Devika [36]'s proof is not rigorous enough. Besides the above two flaws, it has another limitation that it only analyzes the case that the paths in the network are symmetric.

This thesis studies a case of ACO with network routing application using the RRA's pheromone updating rules -- RACO. The RRA pheromone updating rules will be repeated in the next section. Theoretical results show that the pheromone update rate plays a critical role in the convergence behavior of RACO.

5.2 Two-Node, Two-Path Model

Convergence is first analyzed by considering a two-node, two path asymmetric network shown in Fig 5.1.

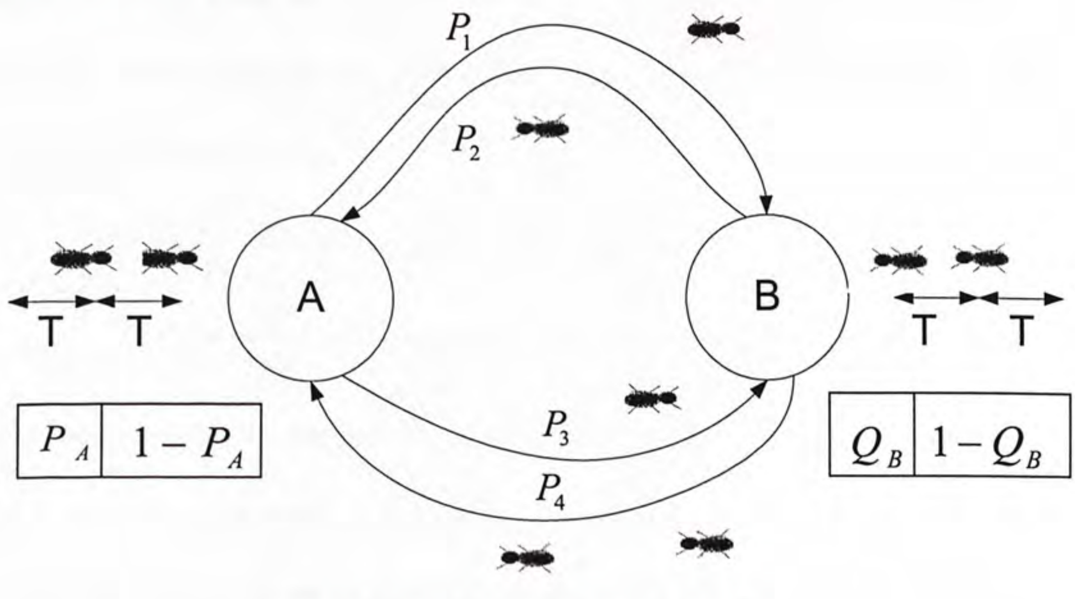

Fig 5.1: Two-Node, Two-Path Model.

This model has two bi-directional paths: the upper path and the lower path. Because the network is asymmetric, one bi-directional path is divided into two unidirectional paths that are presented in the Fig 5.1. For example, the upper path is divided into one unidirectional path from A to B (that is P_1) and another one from B to A (that is P_2). At every regular interval T, two ants enter the system at node A and node B respectively. Every ant chooses the path probabilistically based on the pheromone values at its starting node. Ants staring at node A choose the upper path P_1 with probability P_A and the lower path P_3 with probability $1-P_A$; ants starting at node B choose the upper path P_2 with probability Q_B and the lower path P_4 with probability $1-Q_B$. The costs $\{C_1, C_2, C_3, C_4\}$ that represent the time taken to transverse path $\{P_1, P_2, P_3, P_4\}$ are static through the pheromone update process. Without loss of generality, it is assumed that $C_i \neq C_j$ $(i \neq j,\ i, j \in \{1,2,3,4\})$ and $\max\limits_{i=1,\ldots,4}(c_i)$ >T. When each ant arrives its destination, it updates the destination's pheromone values using the updating rules presented in the chapter 3. For example, there is an ant from node B arriving at node A. The pheromone value in node A will be updated by following rules:

$$P_A' = \begin{cases} I(P_A) = \dfrac{P_A + \Delta p_2}{1+\Delta p_2}, \text{ if the ant comes from the upper path } P_2 & (5.1) \\ D(P_A) = \dfrac{P_A}{1+\Delta p_4}, \text{if the ant comes from the lower path } P_4 & (5.2) \end{cases},$$

where the probability P_A denotes the last pheromone value; the functions $I(P_A)$ and $D(P_A)$ mean the increasing function and decreasing function, respectively; the variable Δp_i (i =2,4) is called as the pheromone update rate of each unidirectional path P_i. Pheromone update rate is defined as follows: suppose that C_A be the minimum delay (cost) from node A to node B and C_{ci} be the delay (cost) carried back from path P_i. Then, $\Delta p_i = \begin{cases} C_A / C_{ci}, \text{ if } C_{ci} \le C_A & (5.3) \\ 0.1(C_A / C_{ci}), \text{ if } C_{ci} > C_A & (5.4) \end{cases}$, where i =2, 4.

Similarly, the updating rules in node B is defined as follows:

$$Q_B^{'} = \begin{cases} I(Q_B) = \dfrac{Q_B + \Delta p_1}{1 + \Delta p_1}, \text{ if the ant comes from the upper path } P_1 & (5.5) \\ D(Q_B) = \dfrac{Q_B}{1 + \Delta p_3}, \text{ if the ant comes from the lower path } P_3 & (5.6) \end{cases},$$

where the probability Q_B is the last pheromone value; the functions $I(Q_B)$ and $D(Q_B)$ mean the increasing function and decreasing function, respectively; the variable Δp_i (i =1,3) is the pheromone update rate of each unidirectional path P_i. Pheromone update rate is defined as follows: suppose C_B be the minimum delay(cost) from node B to node A and C_{ci} be the delay (cost) carried back from path P_i. Then, $\Delta p_i = \begin{cases} C_B / C_{ci}, \text{ if } C_{ci} \le C_B & (5.7) \\ 0.1(C_B / C_{ci}), \text{ if } C_{ci} > C_B & (5.8) \end{cases}$, where i =1, 3.

To facilitate the analysis of convergence property of RRA, a set of notations is defined as follows:

- Let P_A^t, Q_B^t be the pheromone value at the time interval **t** in the node A and B, respectively. Accordingly, P_A^0 and Q_B^0 denotes the initial pheromone value when t = 0.Assume that $P_A^0, Q_B^0 \neq 0,1$. Symbols of P_A and Q_B refer to the generic pheromone value.
- The number of intervals for traversing the paths is defined as: $m_1 = \left\lceil \dfrac{C_2}{T} \right\rceil$, $n_1 = \left\lceil \dfrac{C_4}{T} \right\rceil$, $m_2 = \left\lceil \dfrac{C_1}{T} \right\rceil$, $n_2 = \left\lceil \dfrac{C_3}{T} \right\rceil$. Hence, m_1 and n_1 are the intervals that traverse the upper path and lower path from B to A, respectively; m_2 and n_2 are the intervals that traverse the upper path and lower path from A to B,

respectively.

- The residual length is defined as: $u_1 = m_1 - \frac{C_2}{T}$, $v_1 = n_1 - \frac{C_4}{T}$, $u_2 = m_2 - \frac{C_1}{T}$, $v_2 = n_2 - \frac{C_3}{T}$.

Fig 5.2 illustrates the above definitions.

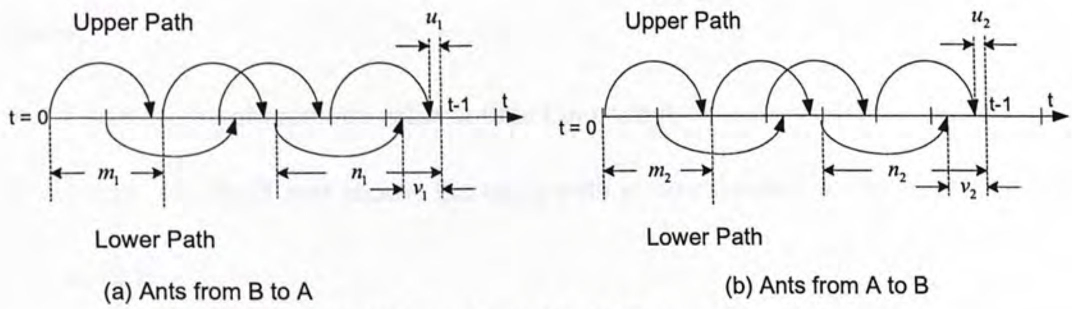

Fig 5.2: Illustration of Ants Traversing the Path.

5.3 The Recursive Pheromone Values

As it has claimed before, the updating behavior of RRA is a non-Markov process that the current state of pheromone value relies on not only the nearest previous state, but also other previous states of pheromone on other nodes as well. So this section will show the relationship between the current state of pheromone value and previous states of pheromone values. Results show that the relationship among different states is nonlinear and thus very complicated.

When an ant arrives at the node A from node B, it updates the pheromone value P_A. So the pheromone value P_A^t depends not only on P_A^{t-1}, but also on the some previous states of pheromone value in the node B. From Fig 5.2 (a), the pheromone value P_A^t depends on the last interval pheromone value P_A^{t-1}, the last m_1 interval pheromone value $Q_B^{t-m_1}$ and the last n_1 interval pheromone value $Q_B^{t-n_1}$. Similarly, the pheromone value Q_B^t depends on the last interval pheromone value Q_B^{t-1}, the last

m_2 interval pheromone value $P_A^{t-m_2}$ and the last n_2 interval pheromone value $P_A^{t-n_2}$.

Total Probability Theorem[62] is used to derive the exact form of P_A^t (Q_B^t). Before applying the Total Probability Theorem, it is necessary to define a set of events:

- Event G: the pheromone value at time t in node A
- Event H_1: Both ants choose the upper path at time interval $t-m_1$ and $t-n_1$ in node B
- Event H_2: One ant chooses the upper path at time interval $t-m_1$ and another ant chooses the lower path $t-n_1$ in node B
- Event H_3: One ant chooses the lower path at time interval $t-m_1$ and another ant chooses the upper path $t-n_1$ in node B
- Event H_4: Both ants choose the lower path at time interval $t-m_1$ and $t-n_1$ in node B

According to the total probability theorem, we have:

$$P(G) = P(H_1)P(G \mid H_1) + P(H_2)P(G \mid H_2) + P(H_3)P(G \mid H_3) + P(H_4)P(G \mid H_4) \quad (5.9)$$

By the different relationship between u_1 and v_1, u_2 and v_2, there are totally four cases to derivate P_A^t (Q_B^t).

The first case is $u_1 < v_1$ and $u_2 < v_2$ illustrated in Fig 5.3.

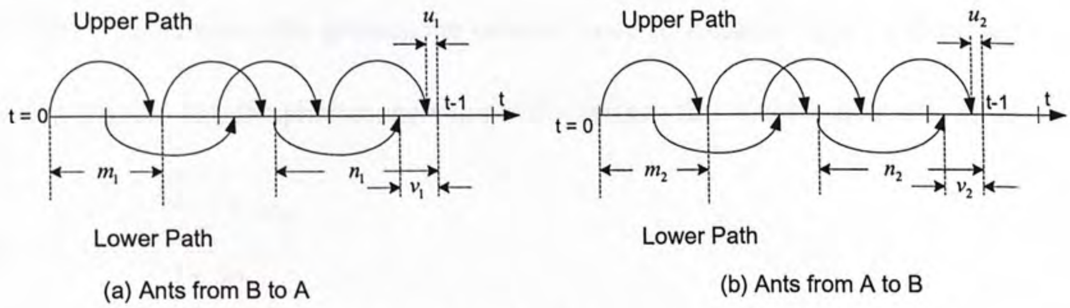

Fig 5.3: The Case of $u_1 < v_1$ and $u_2 < v_2$.

Lemma 5.1: For $u_1 < v_1$ and $u_2 < v_2$, the pheromone values P_A^t and Q_B^t are given as follows:

$$P_A^t = Q_B^{t-m_1} Q_B^{t-n_1} \frac{P_A^{t-1} + \Delta p_2}{1+\Delta p_2} + Q_B^{t-m_1}(1-Q_B^{t-n_1}) \frac{\frac{P_A^{t-1}}{1+\Delta p_4} + \Delta p_2}{1+\Delta p_2} + (1-Q_B^{t-m_1}) Q_B^{t-n_1} P_A^{t-1} + (1-Q_B^{t-m_1})(1-Q_B^{t-n_1}) \frac{P_A^{t-1}}{1+\Delta p_4} \tag{5.10}$$

and

$$Q_B^t = P_A^{t-m_2} P_A^{t-n_2} \frac{Q_B^{t-1} + \Delta p_1}{1+\Delta p_1} + P_A^{t-m_2}(1-P_A^{t-n_2}) \frac{\frac{Q_B^{t-1}}{1+\Delta p_3} + \Delta p_1}{1+\Delta p_1} + (1-P_A^{t-m_2}) P_A^{t-n_2} Q_B^{t-1} + (1-P_A^{t-m_2})(1-P_A^{t-n_2}) \frac{Q_B^{t-1}}{1+\Delta p_3} \tag{5.11}$$

Proof:

The first term on the RHS of equation (5.10) stands for $P(H_1)P(G \mid H_1)$: The probability of both ants choosing the upper path at time interval $t-m_1$ and $t-n_1$ in node B is $Q_B^{t-m_1} Q_B^{t-n_1}$ that denotes $P(H_1)$. Only the ant starting at the time interval $t-m_1$ increases the pheromone value at node A by $\frac{P_A^{t-1} + \Delta p_2}{1+\Delta p_2}$.

The second term on the RHS of equation (5.10) stands for $P(H_2)P(G \mid H_2)$: The probability of one ant choosing the upper path at time interval $t-m_1$ and another ant choosing the lower path $t-n_1$ in node B is $Q_B^{t-m_1}(1-Q_B^{t-n_1})$ that denotes $P(H_2)$.

Both ants will influence the pheromone value in node A. Because $u_1 < v_1$, observing from the Fig 5.3 (a), the pheromone value will decrease first and the increase. Thus,

$$P(G \mid H_2) = \frac{\dfrac{P_A^{t-1}}{1+\Delta p_4} + \Delta p_2}{1+\Delta p_2}.$$

The third term on the RHS of equation (5.10) stands for $P(H_3)P(G \mid H_3)$: The probability of one ant choosing the lower path at time interval $t-m_1$ and another ant choosing the upper path $t-n_1$ in node B is $(1-Q_B^{t-m_1})Q_B^{t-n_1}$ that denotes $P(H_3)$. Both ants will not influence the pheromone value in node A. Thus, $P(G \mid H_3) = P_A^{t-1}$.

The fourth term in equation (5.10) on the RHS stands for $P(H_4)P(G \mid H_4)$: The probability of both ants chooses the lower path at time interval $t-m_1$ and $t-n_1$ in node B is $(1-Q_B^{t-m_1})(1-Q_B^{t-n_1})$ that denotes $P(H_4)$. Only the ant choosing the lower path at time $t-n_1$ will decrease the pheromone value. Thus, $P(G \mid H_4) = \dfrac{P_A^{t-1}}{1+\Delta p_4}$.

By equation (5.9),

$$\begin{aligned} P_A^t = {} & Q_B^{t-m_1} Q_B^{t-n_1} \frac{P_A^{t-1} + \Delta p_2}{1+\Delta p_2} + Q_B^{t-m_1}(1-Q_B^{t-n_1}) \frac{\dfrac{P_A^{t-1}}{1+\Delta p_4} + \Delta p_2}{1+\Delta p_2} + \\ & (1-Q_B^{t-m_1}) Q_B^{t-n_1} P_A^{t-1} + (1-Q_B^{t-m_1})(1-Q_B^{t-n_1}) \frac{P_A^{t-1}}{1+\Delta p_4} \end{aligned}.$$

By the same approach, equation (5.11) can be derived. **Q.E.D**

The second case is $u_1 < v_1$ and $u_2 > v_2$ illustrated in Fig 5.4.

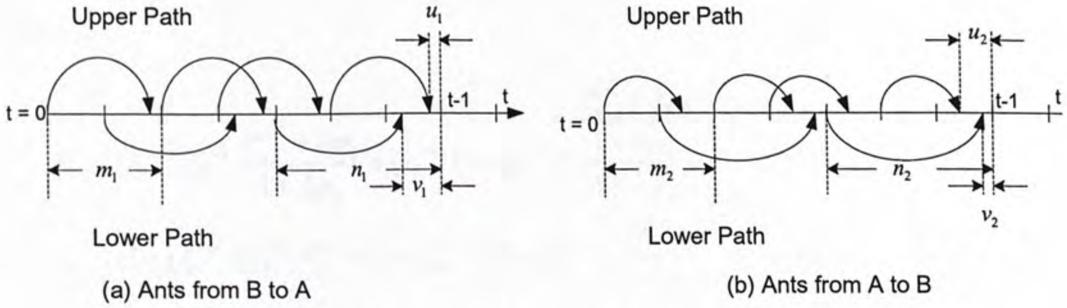

Fig 5.4: The Case of $u_1 < v_1$ and $u_2 > v_2$.

Lemma 5.2: For $u_1 < v_1$ and $u_2 > v_2$, the pheromone values P_A^t and Q_B^t are given as follows:

$$P_A^t = Q_B^{t-m_1} Q_B^{t-n_1} \frac{P_A^{t-1} + \Delta p_2}{1+\Delta p_2} + Q_B^{t-m_1}(1-Q_B^{t-n_1}) \frac{\frac{P_A^{t-1}}{1+\Delta p_4} + \Delta p_2}{1+\Delta p_2} +$$
$$(1-Q_B^{t-m_1}) Q_B^{t-n_1} P_A^{t-1} + (1-Q_B^{t-m_1})(1-Q_B^{t-n_1}) \frac{P_A^{t-1}}{1+\Delta p_4} \tag{5.12}$$

and

$$Q_B^t = P_A^{t-m_2} P_A^{t-n_2} \frac{Q_B^{t-1} + \Delta p_1}{1+\Delta p_1} + P_A^{t-m_2}(1-P_A^{t-n_2}) \frac{\frac{Q_B^{t-1} + \Delta p_1}{1+\Delta p_1}}{1+\Delta p_3} +$$
$$(1-P_A^{t-m_2}) P_A^{t-n_2} Q_B^{t-1} + (1-P_A^{t-m_2})(1-P_A^{t-n_2}) \frac{Q_B^{t-1}}{1+\Delta p_3} \tag{5.13}$$

Proof: The proof is similar with the proof in Lemma 5.1. It is not repeated here.

The third case is $u_1 > v_1$ and $u_2 < v_2$ illustrated in Fig 5.5.

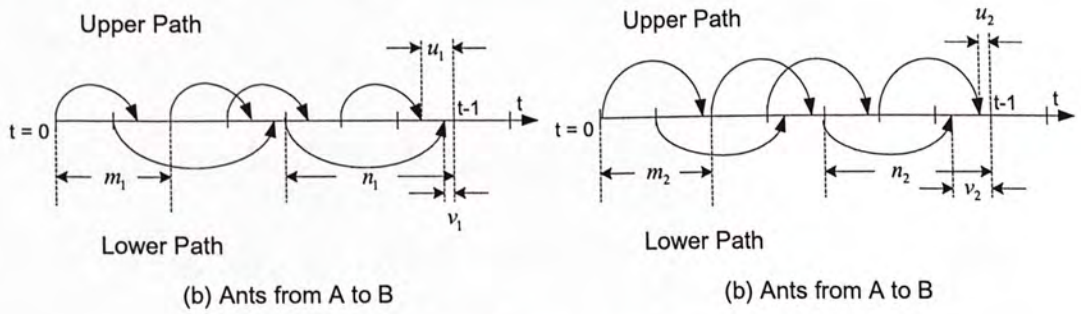

Fig 5.5: The Case of $u_1 > v_1$ and $u_2 < v_2$.

Lemma 5.3: For $u_1 > v_1$ and $u_2 < v_2$, the pheromone values P_A^t and Q_B^t are given as follows:

$$P_A^t = Q_B^{t-m_1} Q_B^{t-n_1} \frac{P_A^{t-1} + \Delta p_2}{1+\Delta p_2} + Q_B^{t-m_1}(1-Q_B^{t-n_1}) \frac{\frac{P_A^{t-1}+\Delta p_2}{1+\Delta p_2}}{1+\Delta p_4} + (1-Q_B^{t-m_1})Q_B^{t-n_1}P_A^{t-1} + (1-Q_B^{t-m_1})(1-Q_B^{t-n_1})\frac{P_A^{t-1}}{1+\Delta p_4} \tag{5.14}$$

and

$$Q_B^t = P_A^{t-m_2} P_A^{t-n_2} \frac{Q_B^{t-1} + \Delta p_1}{1+\Delta p_1} + P_A^{t-m_2}(1-P_A^{t-n_2}) \frac{\frac{Q_B^{t-1}}{1+\Delta p_3}+\Delta p_1}{1+\Delta p_1} + (1-P_A^{t-m_2})P_A^{t-n_2}Q_B^{t-1} + (1-P_A^{t-m_2})(1-P_A^{t-n_2})\frac{Q_B^{t-1}}{1+\Delta p_3} \tag{5.15}$$

Proof: The proof is analogous to the proof in Lemma 5.1. It is not repeated here.

The fourth case is $u_1 > v_1$ and $u_2 > v_2$ illustrated in Fig 5.6.

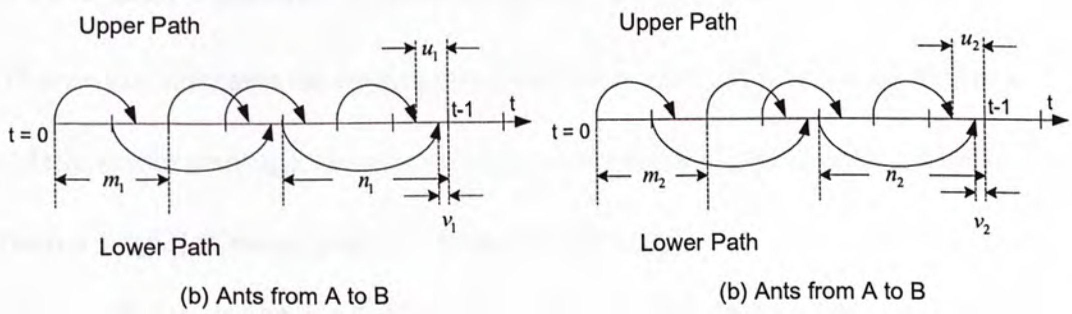

Fig 5.6: The Case of $u_1 > v_1$ **and** $u_2 > v_2$**.**

Lemma 5.4: For $u_1 > v_1$ and $u_2 > v_2$, the pheromone values P_A^t and Q_B^t are given as follows:

$$P_A^t = Q_B^{t-m_1} Q_B^{t-n_1} \frac{P_A^{t-1} + \Delta p_2}{1+\Delta p_2} + Q_B^{t-m_1}(1-Q_B^{t-n_1}) \frac{\frac{P_A^{t-1}+\Delta p_2}{1+\Delta p_2}}{1+\Delta p_4} + (1-Q_B^{t-m_1})Q_B^{t-n_1}P_A^{t-1} + (1-Q_B^{t-m_1})(1-Q_B^{t-n_1})\frac{P_A^{t-1}}{1+\Delta p_4} \tag{5.16}$$

and

$$Q_B^t = P_A^{t-m_2} P_A^{t-n_2} \frac{Q_B^{t-1} + \Delta p_1}{1+\Delta p_1} + P_A^{t-m_2}(1-P_A^{t-n_2}) \frac{\frac{Q_B^{t-1}+\Delta p_1}{1+\Delta p_1}}{1+\Delta p_3} +$$
$$(1-P_A^{t-m_2}) P_A^{t-n_2} Q_B^{t-1} + (1-P_A^{t-m_2})(1-P_A^{t-n_2}) \frac{Q_B^{t-1}}{1+\Delta p_3} \tag{5.17}$$

Proof: The proof is similar with the proof in Lemma 5.1. It is not repeated here.

When the case $u_1 = v_1$ or $u_2 = v_2$ happens, it means that two ants (one from the upper path and the other from the lower path) arrive their destinations simultaneously. It is prescribed that the increase of pheromone value P_A or Q_B does first, and then the decrease action. Therefore, the case $u_1 = v_1$ *and* $u_2 = v_2$ is equivalent to the case $u_1 > v_1$ *and* $u_2 > v_2$ that has been discussed in Lemma 5.4.

5.4 Pheromone Convergence

This section discusses the convergence behavior of RRA. It is necessary to give a clear definition of convergence before we begin our journal of convergence analysis.

Definition 5.1[63] (Convergence in Mathematics):

The number L is the limit of the sequence $\{a_n\}$ if given $\varepsilon > 0$, $a_n \underset{\varepsilon}{\approx} L$ for $n \gg 1$. If such L exists, we say $\{a_n\}$ **converges**, or is *convergent*; otherwise, $\{a_n\}$ **diverges**, or is *divergent*. Another notation for the limit of the sequence $\{a_n\}$ is: $\lim\limits_{n\to\infty}\{a_n\} = L$.

Definition 5.2 (Convergence in RACO): Suppose there are **k** paths between two nodes, the probability of choosing a particular path among these **k** paths at each of these two nodes remains constant in the long run, then the RRA ACO approach is said to be convergent. If in addition, the probability of choosing the shortest path among these **k** paths is one, then the RACO approach is said to converge to the optimal path or optimal solution

Lemma 5.5: Suppose that the RACO approach converges, then the limit points of (P_A^t, Q_B^t) can be $(0,\ 0)$ or $(1,\ 1)$.

Proof: For $u_1 < v_1$ and $u_2 < v_2$, take the limit on both sides of equation (5.10)(5.11) and let $\lim_{t\to\infty} P_A^t = \overline{P}$ and $\lim_{t\to\infty} Q_B^t = \overline{Q}$.

Equations (5.10) and (5.11) become

$$\overline{P} = \overline{Q}^2 \frac{\overline{P}+\Delta p_2}{1+\Delta p_2} + \overline{Q}(1-\overline{Q}) \frac{\dfrac{\overline{P}}{1+\Delta p_4}+\Delta p_2}{1+\Delta p_2} + (1-\overline{Q})\overline{Q}\overline{P} + (1-\overline{Q})^2 \frac{\overline{P}}{1+\Delta p_4} \tag{5.18}$$

$$\overline{Q} = \overline{P}^2 \frac{\overline{Q}+\Delta p_1}{1+\Delta p_1} + \overline{P}(1-\overline{P}) \frac{\dfrac{\overline{Q}}{1+\Delta p_3}+\Delta p_1}{1+\Delta p_1} + (1-\overline{P})\overline{P}\overline{Q} + (1-\overline{P})^2 \frac{\overline{Q}}{1+\Delta p_3} \tag{5.19}$$

Obviously, $\begin{cases}\overline{P}=1\\ \overline{Q}=1\end{cases}$ and $\begin{cases}\overline{P}=0\\ \overline{Q}=0\end{cases}$ are two groups of solutions of equations (5.18)(5.19).

Similarly, for $u_1 < v_1$ and $u_2 > v_2$, $u_1 > v_1$ and $u_2 < v_2$, or $u_1 > v_1$ and $u_2 > v_2$, $(\overline{P},\ \overline{Q})$ can also be $(0,\ 0)$ and $(1,\ 1)$. **Q.E.D.**

Consider the model in Fig 5.1, there are two paths from node **A** to node **B** and another two paths from node **B** to node **A**. Two general cases are classified to analyze the convergence behavior: The first case is the both shortest paths from node **A** to node **B** and from node **B** to node **A** belong to the upper, bi-directional path or the lower one; the other case is these two shortest paths belong to the different bi-directional paths, namely, one belongs to the upper path and the other belongs to the lower path. Without loss of generality, we suppose that case I is $C_1 < C_3$ and $C_2 < C_4$ and case II is $C_1 < C_3$ and $C_2 > C_4$.

5.4.1 Case I: $C_1 < C_3$ and $C_2 < C_4$

When $C_1 < C_3$ and $C_2 < C_4$, by equations (5.3)(5.4)(5.7)(5.8), $\Delta p_1 = 1, \Delta p_2 = 1$, $\Delta p_3 = 0.1\frac{C_1}{C_3}$, $\Delta p_4 = 0.1\frac{C_2}{C_4}$. Let $a = \frac{1}{1+\Delta p_4}$, $b = \frac{1}{1+\Delta p_3}$. We make some preparations for the analysis.

Lemma 5.6: For all values of P_A and Q_B, after one ant brings the shortest delay (cost) from the upper path P_1 or P_2, the updated pheromone value $P_A^{'}$, $Q_B^{'} \geq$ 1/2.

Proof: $P_A^{'} = \frac{P_A + \Delta p_2}{1+\Delta p_2} \overset{\Delta p_2 = 1}{=} \frac{1+P_A}{2} \geq \frac{1}{2}$.

Similarly, $Q_B^{'} \geq \frac{1}{2}$. **Q.E.D**

Lemma 5.7: The maximum decrease of pheromone value at any time is $1 - \frac{1}{1.1}$ (≈ 0.0909).

Proof: $P_A - P_A^{'} = P_A - \frac{P_A}{1+\Delta p_4} = P_A(1 - \frac{1}{1+\Delta p_4})$.

Because $\Delta p_4 < 0.1$, we have $\frac{1}{1+\Delta p_4} > \frac{1}{1.1}$. So $P_A - P_A^{'} \leq 1 - \frac{1}{1+\Delta p_4} < 1 - \frac{1}{1.1}$

The proof for Q_B is the same. **Q.E.D**

Lemma 5.8: In the RACO model presented in Fig 5.1, if $C_1 < C_3$ and $C_2 < C_4$, then the expectation values of P_A and Q_B are bigger than 1/2, that is, $E[P_A], E[Q_B] > \frac{1}{2}$.

Proof: We analyze the case $u_1 < v_1$ and $u_2 < v_2$.

By equation (5.10),

$$P_A^t = Q_B^{t-m_1} Q_B^{t-n_1} \frac{P_A^{t-1}+\Delta p_2}{1+\Delta p_2} + Q_B^{t-m_1}(1-Q_B^{t-n_1}) \frac{\frac{P_A^{t-1}}{1+\Delta p_4}+\Delta p_2}{1+\Delta p_2} +$$

$$(1-Q_B^{t-m_1})Q_B^{t-n_1} P_A^{t-1} + (1-Q_B^{t-m_1})(1-Q_B^{t-n_1}) \frac{P_A^{t-1}}{1+\Delta p_4}$$

$$= Q_B^{t-m_1} Q_B^{t-n_1} \frac{P_A^{t-1}+\Delta p_2}{1+\Delta p_2} - Q_B^{t-m_1} Q_B^{t-n_1} \frac{\frac{P_A^{t-1}}{1+\Delta p_4}+\Delta p_2}{1+\Delta p_2} + Q_B^{t-m_1} \frac{\frac{P_A^{t-1}}{1+\Delta p_4}+\Delta p_2}{1+\Delta p_2}$$

$$(1-Q_B^{t-m_1})Q_B^{t-n_1} P_A^{t-1} + (1-Q_B^{t-m_1})(1-Q_B^{t-n_1}) \frac{P_A^{t-1}}{1+\Delta p_4}$$

Because $P_A^{t-1} > \frac{P_A^{t-1}}{1+\Delta p_4}$, we have $Q_B^{t-m_1} Q_B^{t-n_1} \frac{P_A^{t-1}+\Delta p_2}{1+\Delta p_2} > Q_B^{t-m_1} Q_B^{t-n_1} \frac{\frac{P_A^{t-1}}{1+\Delta p_4}+\Delta p_2}{1+\Delta p_2}$.

Therefore, $P_A^t > Q_B^{t-m_1} \frac{\frac{P_A^{t-1}}{1+\Delta p_4}+\Delta p_2}{1+\Delta p_2} + (1-Q_B^{t-m_1}) P_A^{t-1} (Q_B^{t-n_1} + \frac{1}{1+\Delta p_4} - Q_B^{t-n_1} \frac{1}{1+\Delta p_4})$

Substitute $\Delta p_2 = 1$ and $a = \frac{1}{1+\Delta p_4}$ to the above equation,

we have $P_A^t > Q_B^{t-m_1} \frac{1-aP_A^{t-1}}{2} + aP_A^{t-1}$, where $1 > a > \frac{1}{1.1}$.

If $P_A^{t-1} \geq 0.55$, then $P_A^t > Q_B^{t-m_1} \frac{1-aP_A^{t-1}}{2} + aP_A^{t-1} \geq aP_A^{t-1} > \frac{1}{2}$.

If $0.55 > P_A^{t-1} \geq 0.5$ and $Q_B^{t-m_1} \geq 0.2$, then

$$P_A^t > Q_B^{t-m_1} \frac{1-aP_A^{t-1}}{2} + aP_A^{t-1} \geq 0.2 \frac{1-0.55}{2} + \frac{1}{1.1} * \frac{1}{2} > \frac{1}{2}.$$

By Lemma 5.6 and Lemma 5.7, inequality $P_A^{t-1} \geq 0.55$ or $0.55 > P_A^{t-1} \geq 0.5$ and $Q_B^{t-m_1} \geq 0.2$ can be satisfied at very great chance, so $P_A^t > \frac{1}{2}$ is true at the most time. Another interpretation is that if P_A is uniformly distributed in the domain of [0,1], then $E[P_A] = \frac{1}{2}$. By Lemma 5.6 and Lemma 5.7, P_A is prone to the value that is bigger than 1/2. Thus, $E[P_A] > \frac{1}{2}$.

Similarly, $E[Q_B] > \frac{1}{2}$. The analysis for the case of $u_1 < v_1$ and $u_2 > v_2$, $u_1 > v_1$ and $u_2 < v_2$ or $u_1 > v_1$ and $u_2 > v_2$ is the same.

Therefore, in the model shown in Fig 5.1, if $C_1 < C_3$ and $C_2 < C_4$, then $E[P_A], E[Q_B] > \frac{1}{2}$. **Q.E.D.**

Theorem 5.1: In the RACO model presented in Fig 5.1, if the path is the shortest path connecting node A with node B bi-directionally, the approach will converge to the optimal path in the sense that the probability of choosing the bi-directional, shortest path equals to one in the long run.

Proof: The case of $C_1 < C_3$ and $C_2 < C_4$ means the upper path is the shortest path connecting node A and node B bi-directionally. So $(\bar{P}, \bar{Q})$=(1,1) denotes the RACO approach converges to the optimal path. In the following we will prove $(\bar{P}, \bar{Q})$ will be definitely driven to (1,1).

By Lemma 5.5, if the RACO approach converges, then $(\bar{P}, \bar{Q})$=(1,1). By Lemma 5.8, $E[P_A], E[Q_B] > \frac{1}{2}$ means that during the pheromone update process, the number of increasing the pheromone values of P_A and Q_B is bigger than that of decreasing them. Besides, at each increasing or decreasing time, the amount of increase is always much bigger than the amount of decrease by Lemma 5.6 and Lemma 5.7. Therefore, in the long run, P_A, Q_B will converge to (1,1). The RACO approach converges to the optimal path. **Q.E.D.**

5.4.2 Case II: $C_1 < C_3$ and $C_2 > C_4$.

The condition of $C_1 < C_3$ and $C_2 > C_4$ implies that the shortest path from node A to node B and the shortest path from node B to node A belong to the different bi-direction paths. By equations (5.3)(5.4)(5.7)(5.8), $\Delta p_1 = 0.1\frac{C_4}{C_2}, \Delta p_2 = 1$, $\Delta p_3 = 1$, $\Delta p_4 = 0.1\frac{C_2}{C_4}$. Let $a = \frac{1}{1+\Delta p_4}$, $b = \frac{1}{1+\Delta p_1}$ ($a \in (\frac{1}{1.1}, 1)$ and $b \in (\frac{1}{1.1}, 1)$).

We analyze the case $u_1 < v_1$ and $u_2 < v_2$ first.

For $u_1 < v_1$ and $u_2 < v_2$, substitute $\Delta p_2 = 1$, $\Delta p_3 = 1$, $a = \frac{1}{1+\Delta p_4}$ and $b = \frac{1}{1+\Delta p_1}$ in equations (5.18)(5.19).

Equations (5.18)(5.19) becomes

$$\begin{cases} \overline{P}[(1-a)\overline{Q}^2 + (3a-2)Q + 2 - 2a] = \overline{Q} & (5.20) \\ \overline{Q}[(1-b)\overline{P}^2 - b\overline{P} = 1] = 2(1-b)\overline{P} & (5.21) \end{cases}$$

By equation (5.21), $\overline{Q} = \frac{2(1-b)\overline{P}}{(1-b)\overline{P}^2 - b\overline{P} + 1}$.

Substitute $\overline{Q} = \frac{2(1-b)\overline{P}}{(1-b)\overline{P}^2 - b\overline{P} + 1}$ into equation (5.20), equation (5.20) becomes an equation of five degree:

$$\overline{P}[(1-a)(\frac{2(1-b)\overline{P}}{(1-b)\overline{P}^2 - b\overline{P} + 1})^2 + (3a-2)(\frac{2(1-b)\overline{P}}{(1-b)\overline{P}^2 - b\overline{P} + 1}) + 2 - 2a] = \frac{2(1-b)\overline{P}}{(1-b)\overline{P}^2 - b\overline{P} + 1} \quad (5.22)$$

According to Abel's theorem[64], equations higher than fourth degree are incapable of algebraic solution in terms of a finite number of additions, subtractions, multiplications, divisions, and root extractions. Fortunately, equation (5.22) can be reduced to a quartic equation by deducting $\overline{P}$ on the both sides of equation (5.22). Thus, equation (5.22) is solvable and so equations (5.20)(5.21) are.

By solving equations (5.20)(5.21), we have three groups of real solutions and two groups of complex solutions. Two groups of complex solutions can be discarded immediately by checking whether the values of **a** and **b** that make the imaginary part be zero are in the domain of $(\frac{1}{1.1},1)$. These three groups of real solution are

$\begin{cases}\overline{P}=1\\ \overline{Q}=1\end{cases}$, $\begin{cases}\overline{P}=0\\ \overline{Q}=0\end{cases}$, $\begin{cases}\overline{P}=f_1(a,b)\\ \overline{Q}=f_2(a,b)\end{cases}$, where $f_1(a,b)$ and $f_2(a,b)$ are two real functions in terms of **a** and **b** (Please refer to the Appendix 1 for details) .

Lemma 5.9: If $C_1<C_3$ and $C_2>C_4$, then (P_A,Q_B) can not converge to the limit points (0,0) or (1,1).

Proof: By Lemma 5.6 and assumption $P_A^0,Q_B^0\neq 0,1$, there must be an ant coming from path P_3 to update the pheromone value in node B, Q_B will be smaller than 1/2 because this time the shortest path from node B to node A is P_4. And by Lemma 5.6 and Lemma 5.7, $E[Q_B]<\frac{1}{2}$ and $E[P_A]>\frac{1}{2}$. Thus, (P_A,Q_B) can not converge to the limit points (0,0) or (1,1). **Q.E.D.**

Now, the problem is that whether the pheromone value (P_A,Q_B) can converge to ($\overline{P},\overline{Q}$), where $\overline{P},\overline{Q}\in R$ and $\overline{P},\overline{Q}\in(0,1)$. So, the problem becomes: for $a\in(\frac{1}{1.1},1)$ and $b\in(\frac{1}{1.1},1)$, are $f_1(a,b)$ and $f_2(a,b)$ in the domain between 0 and 1?

Lemma 5.10: For $a\in(\frac{1}{1.1},1)$ and $b\in(\frac{1}{1.1},1)$, $f_1(a,b)$ and $f_2(a,b)$ are not in the domain between 0 and 1.

Proof: The solution $f_1(a,b)$ is analyzed at first. The form of $f_1(a,b)$ is very complicated and tedious and here we just list a part of the $f_1(a,b)$. $f_1(a,b)$ =

1/6/(-b+a*b+1-a)***(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12***(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+…….** (5.23)

Equation (5.23) seems to be astonishing and insolvable at first glance. However, by observation, a piece of equation (5.23) is undertaken a square root computation. So the problem can be further reduced to two steps:

Step 1: To find the domain Ω of **a** and **b** that makes that piece be bigger than 0.

Step 2: To validate whether $f_1(a,b)$ is in the domain of (0,1) subject to $a,b \in \Omega$.

Step 1: Define the piece as the function g(a,b):

$$g(a,b) = \text{-}12a^4b\text{+}9a^4\text{-}36a^3b^2\text{+}36a^3b\text{-}6a^3\text{-}36a^2b^3\text{+}378a^2b^2\text{-}450a^2b\text{+} \\ 141a^2\text{-}12b^4a\text{+}36ab^3\text{-}450ab^2\text{+}606ab\text{-}216a\text{+}9b^4\text{-}6b^3\text{+}141b^2\text{-}216b\text{+}84 \qquad 5.24.$$

To find the domain Ω of **a** and **b** that makes g(a,b) be bigger than 0, it is first to find the value of **a** and **b** such that g(a,b) achieves its maximum value. If the maximum value is smaller than 0, then there is no such a domain Ω of **a** and **b** that makes g(a,b) be bigger than 0. If the maximum value is no less than 0, the domain Ω should be the neighborhood of the point (a^*,b^*) because the g(a,b) is a continuous and smooth function (This is a straightforward result from differential geometry), where $(a^*,b^*) = \arg m\text{ax}(g(a,b))$.

To find the maximum value of g(a,b), the problem is further transformed to an optimization problem:

$$\begin{aligned} \min\ & \text{-}g(a,b) = -(\text{-}12a^4b\text{+}9a^4\text{-}36a^3b^2\text{+}36a^3b\text{-}6a^3\text{-}36a^2b^3\text{+}378a^2b^2\text{-}450a^2b\text{+} \\ & 141a^2\text{-}12b^4a\text{+}36ab^3\text{-}450ab^2\text{+}606ab\text{-}216a\text{+}9b^4\text{-}6b^3\text{+}141b^2\text{-}216b\text{+}84) \\ & \text{subject to: } \frac{1}{1.1} < a < 1, \frac{1}{1.1} < b < 1 \end{aligned} \qquad (5.25)$$

Because g(a,b) is a non-linear function, the minimization of g(a,b) is a non-linear programming problem. A number of non-linear programming methods are used to solve this problem, such as interior-reflective Newton method [66][67]and Nelder and

Mead simplex method[65]. Among these methods, Nelder and Mead simplex method achieves the best performance. Nelder and Mead simplex method should not be confused with simplex method of linear programming that enjoys considerable popularity. By Nelder and Mead simplex method, when a=b=0.9999, -g(a,b) achieves its minimum value -8.5265e-014. Further analysis shows that only when $a \in (0.9996,1)$ and $b \in (0.9996,1)$, g(a,b) is bigger than 0 (Please refer to Appendix 2 for details).

Step 2, When $a \in (0.9996,1)$ and $b \in (0.9996,1)$, $f_1(a,b)$ has a real value that exceeds the domain of [0,1] (Please refer to Appendix 2 for details).

So, for $a \in (1,\frac{1}{1.1})$ and $b \in (1,\frac{1}{1.1})$, there are no values of **a** and **b** such that $f_1(a,b) \in (0,1)$. **Q.E.D.**

By Lemma 5.9, the group of solution $\begin{cases} \overline{P} = f_1(a,b) \\ \overline{Q} = f_2(a,b) \end{cases}$ is discarded.

Corollary 5.1: For $u_1 > v_1$ and $u_2 < v_2$, $u_1 < v_1$ and $u_2 > v_2$, or $u_1 > v_1$ and $u_2 > v_2$, when $C_1 < C_3$ and $C_2 > C_4$, the systems of equations (5.12) (5.13), (5.14) (5.15), and (5.16) (5.17) do not have real value solutions that are between 0 and 1, exclusively.

Proof: The proof is similar with the Lemma 5.9 and 5.10. It is not repeated here.

Theorem 5.2: In the RACO presented in Fig 5.1, if the shortest, unidirectional path from node **A** to node **B** and the shortest, unidirectional path from node **B** to node **A** do not belong to the same bi-directional path, the RACO approach do not converge.

Proof: By the result of Lemma 5.9, (P_A, Q_B) can not converge to (0,0) or (1,1). By the result of Lemma 5.10 and Corollary 5.1, P_A and Q_B can not converge to any value between 0 and 1 simultaneously. Thus, the approach will not converge.

Q.E.D.

The property of non-convergence is not desirable. To make it be convergent, it is needed to make some modifications on the pheromone update rate.

In the model presented in Fig 5.1, there is an ant from node **B** arriving at node **A**. The pheromone value in node **A** will be updated as follows:

$$P_A^{'} = \begin{cases} I(P_A) = \dfrac{P_A + \Delta p_2}{1 + \Delta p_2}, \text{ if the ant comes from the upper path } P_2 & (5.1) \\ D(P_A) = \dfrac{P_A}{1 + \Delta p_4}, \text{if the ant comes from the lower path } P_4 & (5.2) \end{cases},$$

where the probability P_A denotes the last pheromone value; the functions $I(P_A)$ and $D(P_A)$ mean the increasing function and decreasing function, respectively; the variable Δp_i (i =2,4) is called as the pheromone update rate of each unidirectional path P_i. Pheromone update rate is defined as follows: Suppose that C_A be the minimum delay from node **A** to node **B** and C_{ci} be the delay carried back from path P_i. Then,

$$\Delta p_i = \begin{cases} C_A / C_{ci}, \text{ if } C_{ci} \le C_A & (5.26) \\ 0, \text{ if } C_{ci} > C_A & (5.27) \end{cases}, \text{ where i =2, 4.}$$

Similarly, the updating rules in node **B** is defined as follows:

$$Q_B^{'} = \begin{cases} I(Q_B) = \dfrac{Q_B + \Delta p_1}{1 + \Delta p_1}, \text{ if the ant comes from the upper path } P_1 & (5.5) \\ D(Q_B) = \dfrac{Q_B}{1 + \Delta p_3}, \text{ if the ant comes from the lower path } P_2 & (5.6) \end{cases},$$

where the probability Q_B is the last pheromone value; the functions $I(Q_B)$ and $D(Q_B)$ mean the increasing function and decreasing function, respectively; the variable Δp_i (i =1,3) is the pheromone update rate of each unidirectional path P_i.

Pheromone update rate is defined as follows: Suppose that C_B be the minimum delay from node **B** to node **A** and C_{ci} be the delay carried back from path P_i. Then,

$$\Delta p_i = \begin{cases} C_B / C_{ci}, \text{ if } C_{ci} \le C_B & (5.28) \\ 0, \quad\quad \text{if } C_{ci} > C_B & (5.29) \end{cases}, \text{ where i =1, 3.}$$

Besides these modifications of pheromone update rules, an additional rule of ants' behavior is added. The rule is that in the RACO model presented in Fig 5.1, if the pheromone value at one node converges, the RRA may choose the paths randomly at the node rather according to the pheromone values. This rule is called as Random Walk Rule and the RACO approach with new pheromone update rules and the Random Walk Rule is called as modified RACO approach.

Lemma 5.11: In the RACO model presented in Fig 5.1, by the new pheromone update rules shown in (5.26)(5.27)(5.28)(5.29), when $C_1 < C_3$ and $C_2 > C_4$, P_A^t is non-decreasing while Q_B^t is non-increasing.

Proof: The new recursive relation of pheromone value is:

$$\begin{cases} P_A^t = Q_B^{t-m_1} \dfrac{P_A^{t-1} + \Delta p_2}{1+\Delta p_2} + (1 - Q_B^{t-m_1}) P_A^{t-1} & (5.30) \\ Q_B^t = P_A^{t-n_2} \dfrac{Q_B^{t-1}}{1+\Delta p_3} + (1 - P_A^{t-n_2}) Q_B^{t-1} & (5.31) \end{cases}$$

By equation (5.30),

$$\begin{aligned} P_A^t - P_A^{t-1} &= Q_B^{t-m_1} \left(\frac{P_A^{t-1} + \Delta p_2}{1+\Delta p_2} - P_A^{t-1} \right) \\ &= \frac{\Delta p_2 (1 - P_A^{t-1}) Q_B^{t-m_1}}{1+\Delta p_2} \end{aligned}, \quad (5.32)$$

Because $\Delta p_2 > 0$ and $1 \ge P_A^{t-1} \ge 0$, $1 \ge Q_B^{t-m_1} \ge 0$, we have $P_A^t \ge P_A^{t-1}$.

By similar analysis, we have $Q_B^t \le Q_B^{t-1}$. **Q.E.D**

Theorem 5.3. In the RACO model presented in Fig 5.1, when $C_1 < C_3$ and $C_2 > C_4$, the modified RACO approach converges to optimal path.

Proof: In equations (5.32), only when $P_A^{t-1} = 1$ or $Q_B^{t-m_1} = 0$, the equation of $P_A^t = P_A^{t-1}$ holds. If $P_A^{t-1} = 1$ that means the pheromone value in node A converges to the optimal path, by Random Walk Rule, there are still some RRAs choosing P_3 to decrease Q_B. So eventually, the pheromone value Q_B converges to 0. If $Q_B^{t-m_1} = 0$ that means the pheromone value in node B converges to the optimal path, there are still some RRAs choosing P_2 to increase P_A. So eventually, the pheromone value P_A converges to 1. Therefore, the modified RACO approach converges to the optimal path. **Q.E.D.**

Corollary 5.2. The pheromone update rate plays a critical role in the convergence behavior of RACO.

Proof: It is a straightforward result by Theorem 5.2 and Theorem 5.3.

5.5 General Models

5.5.1 Two – Node, N – Path Model

A general case of the model presented in Fig 5.1 is that there are N paths (N>2) between two nodes. This model is illustrated in Fig 5.7. Each path in the model is bi-directional. We assume that the number of paths from node **A** to node **B** is equal to the number of paths from node **B** to node **A**

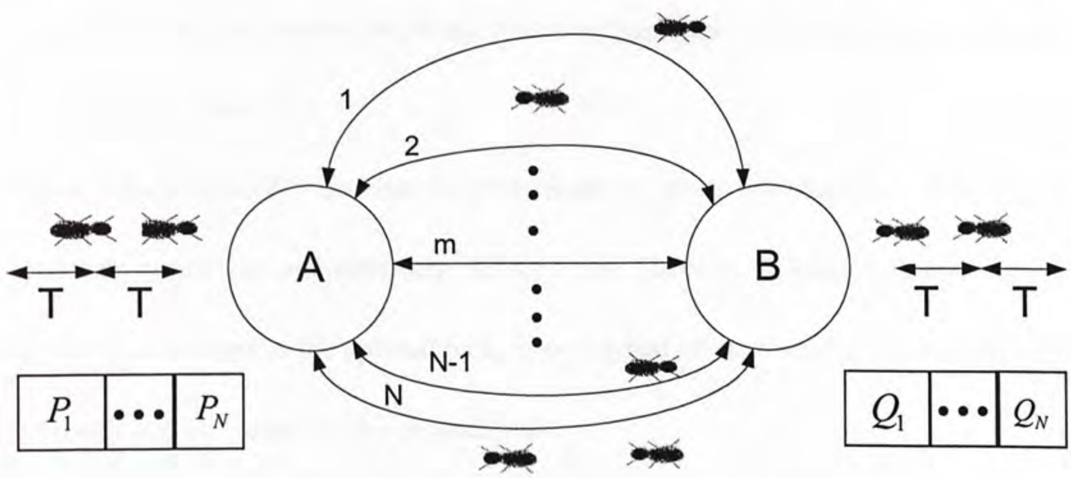

Fig 5.7: Two-Node, N-Path Model.

Corollary 5.3: In a network model presented in Fig 5.7, if the shortest, unidirectional path from node **A** to node **B** and the shortest, unidirectional path from node **B** to node **A** belong to one bi-directional path, the RACO approach converges to the optimal paths.

Proof: Mathematical induction is used to prove this corollary. If N=1, the approach converges automatically. If N = 2, by Theorem 5.1,the RACO approach converges to the optimal path. Suppose that N = k, the RACO approach still converges to the optimal path.

Consider N = k+1. Because N = k, the RACO approach converges to the optimal path. Suppose the optimal path is path **m**. Consider the path **m** and the path **k+1**, it is again a two-path model. If the bidirectional costs of path **m** are bigger than those of path **k+1**, by Theorem 5.1, the RACO approach converges to path **k+1**; otherwise, the RACO approach still converges to path **m** by Theorem 5.1. **Q.E.D.**

Corollary 5.4. In a network model presented in Fig 5.7, if the shortest, unidirectional path from node **A** to node **B** and the shortest, unidirectional path from node **B** to node

A do not belong to one bi-directional path, the modified RACO approach converges to the optimal paths.

Proof: Mathematical induction is again used to prove this theorem. If N=1, the approach converges automatically. If N=2, by Theorem 5.3,the modified RACO approach converges to the optimal path. Suppose that when N=k, the modified RACO approach still converges to the optimal path.

Consider N = k+1. Because when N = k, the modified RACO approach converges to two unidirectional optimal path. Suppose these two unidirectional optimal paths are path **m1** (from node **A** to node **B**) and **m2** (from node **B** to node **A**). Consider the unidirectional paths **m1, m2** and the bidirectional path **k+1**. If the bidirectional costs of path **K+1** are smaller than those of path **m1** and **m2**, by Theorem 5.1, the RACO approach converges to path **k+1**. If the cost of unidirectional path from node **A** to node **B** in path **K+1** is smaller than that of **m1**, by Theorem 5.3, the modified RACO approach still converges to path **m1**; otherwise the modified RACO approach will converge to the unidirectional path from node **A** to node **B** in path **K+1** by Theorem 5.3. The analysis for **m2** is the same as that of **m1.** Therefore, the modified RACO approach converges to optimal paths. **Q.E.D.**

5.5.2 M – Node, N(i.j) – Path Model

The M-node, N(i,j) – Path model means the network has M (M>2) nodes, and from node **i** to **j** there are N(i,j) paths , where $N(i,j) \in N, N(i,j) \geq 2$.

Remark 5.1. In a network with M nodes and between each pair of nodes there are more than two paths between them, if the shortest, unidirectional path from one node to another and the shortest, unidirectional path connecting the same pair of nodes (in the reverse direction) belong to one bi-directional path, the RACO approach converges to the optimal paths.

Discussion: First, it is straightforward the M-node, N(i,j)-path model can be decomposed into at most $\binom{M}{2}$ two –node, N(i,j)-Path models. By Corollary 5.3, each two-node, N-path model converges to the optimal path. Thus, the M-node, N(i,j)-path model also can converge to the optimal paths.

Remark 5.2. In a network with M nodes and between each pair of nodes there are more than two paths between them, if the shortest, unidirectional path from one node to another and the shortest, unidirectional path connecting the same pair of nodes (in the reverse direction) do not belong to one bi-directional path, the modified RACO approach converges to the optimal paths.

Discussion: The discussion is similar with that of Theorem 5.4. It is not repeated here.

5.6. Conclusion and Discussion

This chapter analyzes the convergence behavior of the RACO, one variant of ACO based routing. The result shows that when shortest, unidirectional path from one node to another and the shortest, unidirectional path connecting the same pair of nodes (in the reverse direction) belong to one bi-directional path, the RACO approach converges to the optimal path.

The result also reveals that when the shortest, unidirectional path from node A to node B and the shortest, unidirectional path connecting the same pair of nodes (in the reverse direction) do not belong to one bi-directional path, the approach may not converge. In this case, some modifications to the pheromone update rate are made and Random Walk Rule is included. After these changes, when the shortest paths between two nodes do not belong to the same bi-directional path, the modified RACO approach will converge to optimal paths. This result also tells us that the pheromone update rate plays a critical role in the convergence behavior of RACO approach.

To look for design methodology is the third part of any theoretical analysis (The first part is to formulate a sound mathematical model and the second part is to look for tools to analyze this model.). If two shortest paths connecting pairs of nodes that are in different direction belong to the same bi-directional path, the RACO routing approach can be applied. If the condition is not satisfied, the modified RACO routing approach can be used.

Although the model presented in this chapter and following analysis are for RACO routing, this kind of model and the way of analysis seem also applicable to convergence analysis of other ACO routing approaches, such as AntNet [2][22][23][38][39]. However, more work needs to done to substantiate this claim.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis proposes a new ACO-based routing, Reactive Ant Colony Optimization (RACO), for the packet-switched communication networks. RACO is simple: it does not involve using the second-derivative computation of objective function and the inverse computation of objective function's Hessian matrix that are used in optimal routing, and it does not flood routing information as some traditional routing algorithms (LS and DV) do. RACO is distributed: all nodes in the network compute the routing table based on local information; there is no central controller to coordinate the nodes and ants. RACO is efficient: it is shown by simulation results RACO outperforms some competing routing algorithms in terms of throughput and the average end-to-end delay when the input traffic load is high. RACO is provably convergent: it is demonstrated by the theoretical analysis RACO converge to the optimal path under some network conditions.

The contributions of this thesis can be summarized as follows:

(1) A survey about previous ACO routing algorithms (Chapter 2, Section 1);

(2) A comparative study of ACO routing algorithms with shortest path routing algorithms (Chapter 2, Section 2) and optimal routing (Chapter 2, Section 3);

(3) A new ACO routing approach, RACO, has been designed. (Chapter 3);

(4) An experimental testbed of packet-level simulation has been implemented to comparative evaluation of RACO with one representative of shortest path routing algorithms and two previous ACO routing approaches. (Chapter 4);

(5) The convergence property of RACO has been analyzed. (Chapter 5).

6.2 Future Work

Although RACO routing approach has many merits as presented in the previous sections, it is still a long way to make it be an applicable approach in real networks. There are at least two lines for future work: interplaying with TCP, the analysis of convergence rate.

Since a lot of applications, such as HTTP, FTP and telnet, use TCP as their underlying transport protocol, it appears necessary to consider the interplaying between the RACO routing approaches with TCP. Unlike UPD used in the simulation, TCP has a number of components may greatly influence the routing behavior, such as data retransmission, flow control and congestion control. None of existing ACO based routing even considers this issue. To make RACO be an applicable routing algorithm in the real network, Internet for example, the interplaying with TCP is compulsory to be considered.

Convergence and the rate of convergence are two vital issues for all optimization methods. Convergence analysis refers to whether an optimization method will achieve its optimal value, while the rate of convergence analyzes how fast the optimization method will converge to its optimal value. The convergence property of RACO approach has been addressed in this work. However, the rate of convergence of

RACO approach still remains an open problem. There are several candidate approaches towards quantifying the rate of convergence of an optimization method: (1) *Computational complexity approach.* This method tries to estimate the number of elementary operations needed by a given method to find an optimal solution exactly or within ε-tolerance. (2) *Informational complexity approach or time complexity approach.* This approach estimates the number of function evaluations needed to find an exact or approximate optimal solution. One successful application of this method is [48], in that Wei[48] computed the complexity of gradient projection method for optimal routing in data networks. Nemirovsky and Yudin[73] discussed this approach in detail. (3) *Local Search.* This approach focuses on the local behavior of the method in a neighborhood of an optimal solution. Among these candidate methods, time complexity method seems to be the most promising method to analyze the rate of convergence of RACO approach.

In summary, RACO routing approach, though has already achieves some favorable results both in simulation and theoretical analysis, still has a lot of room for improvements to become an applicable routing protocol in today's communication network, such as Internet.

Reference

[1] Douglas E. Comer, "Computer Networks and Internets", Prentice Hall, 2001

[2] Gianni Di Caro, Marco Dorigo, "AntNet: Distributed Stimergetic Control for communications Networks", *Journal of Artificial Intelligence Research*, 1998

[3] Stone, P., & Veloso, M.M, "Multi-agent systems: A survey from a machine learning perspective. Tech. Rep. CUM-CS-97-193, Carnegie Mellon University, Pittsburgh, PA, 1996.

[4] R. Beckers, J.L. Deneubourg, "*Trails and U-turns in the selection of the shortest path by the ant Lasius Niger*", Journal of theoretical Biology, 159, 397-415: 1992

[5] E. Bonabeau, M. Dorigo, G. Theraulaz, " Inspiration for optimization from social insect behavior", Nature, 406:39-42, 6 July 2000.

[6] M. Dorigo, Gianna Di Caro, L. M. Gambardella, "Ant Algorithm for Discrete Optimization", *Artificial Life*, 5(2): 137-172, 1999.

[7] Dorigo M., V. Maniezzo & A. Colorni, "The Ant System: Optimization by a Colony of Cooperating Agents". *IEEE Transactions on Systems, Man, and Cybernetics-Part B,* 26(1):29-41, 1996

[8] Colorni A., M.Dorigo, F.Maffioli, V. Maniezzo, G. Righini, M. Trubian. "Heuristics from Nature for Hard Combinatorial Problems." *International Transactions in Operational Research,* 3(1):1-21, 1996

[9] Dorigo M. & L.M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem." *IEEE Transactions on Evolutionary Computation,* 1(1):53-66, 1997

[10] Dorigo M. & L.M. Gambardella (1997). "Ant Colonies for the Traveling Salesman Problem", *BioSystems,* 43:73-81. Also Tecnical Report TR/IRIDIA/1996-3, IRIDIA, Université Libre de Bruxelles

[11] Bullnheimer B., R. F. Hartl & C. Strauss (1999), "A New Rank Based Version of the Ant System: A Computational Study", *Central European Journal for Operations Research and Economics,* 7(1):25-38, 1999

[12] Gambardella L. M. and M. Dorigo, "HAS-SOP: An Hybrid Ant System for the Sequential Ordering Problem" *Tech. Rep. No. IDSIA 97-11,* IDSIA, Lugano,

Switzerland, 1997

[13]Gambardella L. M., E. Taillard and M. Dorigo, "Ant Colonies for the Quadratic Assignment Problem" *Journal of the Operational Research Society,* 50:167-176, 1999

[14]Maniezzo V. and A. Colorni, "The Ant System Applied to the Quadratic Assignment Problem", *IEEE Transactions on Knowledge and Data Engineering,* Volume: 11 Issue: 5 , Sep/Oct 1999: 769 -778.

[15]Stützle T. and M. Dorigo, "ACO Algorithms for the Quadratic Assignment Problem", In D. Corne, M. Dorigo and F. Glover, editors, *New Ideas in Optimization,* McGraw-Hill, 1999

[16]Bullnheimer B., R.F. Hartl and C. Strauss, "Applying the Ant System to the Vehicle Routing Problem", In: Voss S., Martello S., Osman I.H., Roucairol C. (eds.), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization,* Kluwer:Boston, 1999

[17]Bullnheimer B, "Ant Colony Optimization in Vehicle Routing", Doctoral thesis, University of Vienna, January 1999

[18]Colorni A., M. Dorigo, V. Maniezzo and M. Trubian, "Ant system for Job-shop Scheduling", *JORBEL - Belgian Journal of Operations Research, Statistics and Computer Science,* 34(1):39-53, 1994

[19]Costa D. and A. Hertz, "Ants Can Color Graphs", *Journal of the Operational Research Society,* 48, 295-305, 1997

[20]Schoonderwoerd R., O. Holland, J. Bruten and L. Rothkrantz, "Ant-based Load Balancing in Telecommunications Networks", *Adaptive Behavior,* 5(2):169-207, 1996

[21]Schoonderwoerd R., O. Holland and J. Bruten, "Ant-like Agents for Load Balancing in Telecommunications Networks", *Proceedings of Agents'97, Marina del Rey, CA, ACM Press,* 209-216, 1997

[22]Di Caro G. and M. Dorigo, "Mobile Agents for Adaptive Routing", *Proceedings of the 31st Hawaii International Conference on System,* IEEE Computer Society Press, Los Alamitos, CA, 74-83, 1998

[23]Di Caro G. and M. Dorigo, "AntNet: A Mobile Agents Approach to Adaptive Routing". *Tech. Rep. IRIDIA/97-12,* Université Libre de Bruxelles, Belgium, 1997

[24] Navarro Varela G. and M.C. Sinclair, Ant Colony "Optimization for Virtual-Wavelength-Path Routing and Wavelength Allocation", *Proceedings of the Congress on Evolutionary Computation (CEC'99),* Washington DC, USA, July 1999.

[25] http://www.isi.edu/nsnam/ns/index.html

[26] Bellman, R.E., "Dynamic Programming", Princeton University Press, Princeton, N.J., 1957

[27] RFC 2453

[28] Dijkstra E., "A note on two problems in connexion with graphs, *Numerische Mathematik* 1", 269-271, 1959

[29] RFC 2328

[30] Ruud Schoonderwoerd, Owen Holland, Janet Bruten, and Leon Rothkrantz, "Ants for Load Balancing in Telecommunication Networks", Hewlett Packard Laboratories Bristol, *Technical Report HPL-96-35*, 1996

[31] Ruud Schoonderwoerd, O. E. Holland, J. Bruten and L. Rothkrantz, "Ant-based load balancing in telecommunications networks", Hewlett Packard Laboratories *Bristol Technical Report HPL-96-76*, May 21, 1996.

[32] Ruud Schoonderwoerd, Owen Holland, "Minimal agents for communications networks routing: the social insect paradigm", *in Software Agents for Future Communication Systems*, Hayzeldean A.L.G. and Bingham J., Ed. Springer-Verlag 1999

[33] Ruud Schoonderwoerd, "Collective Intelligence for Network Control", M.S thesis, Deft University, Netherland, 1996

[34] S. Appleby, S. Steward, "Mobile software agents for control in telecommunication networks, *BT Technology Journal*, vol. 12, no. 2, 1994

[35] E. Bonabeau, F. Hénaux, S. Guérin, D. Snyer, P. Kuntz and G. Théraulaz, "Routing in Telecommunications Networks with Ant-Like Agents", *in Proceedings Intelligent Agents for Telecommunications Applications*, IATA '98, Berlin, Springer-Verlag, 1998.

[36] Devika. Subramanian, P. Druschel, J. Chen, "Ants and Reinforcement Learning: A Case Study in Routing in Dynamic Networks", *In Proceedings 1997 International Joint Conference on Artificial Intelligence,* IJCAI-97, Palo Alto, CA:

Morgan Kaufmann, 1997, p. 832-838.

[37] Martin Heusse, Dominique Snyers, Sylvain Guérin and Pascale Kuntz, "Adaptive Agent-Driven Routing and Load Balancing in Communication Networks", Technical Report RR-98001-IASC, ENST Bretagne, Brest, France 1998

[38] Gianni Di Caro and Marco Dorigo, "Ant Colonies for Adaptive Routing in Packet-switched Communications Networks", *In Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature*, Amsterdam, Holland, September 27-30, 1998

[39] Gianni Di Caro and Marco Dorigo, "An Adaptive Multi-agent Routing Algorithm Inspired by Ants Behavior", in *Proceedings of the 5th Annual Australasian Conference on Parallel and Real-Time Systems*, PART98, Springer-Verlag, 1998, P. 261-272.

[40] Gianni Di Caro and Marco Dorigo, "Two Ant Colony Algorithms for Best-Effort Routing in Datagram Networks", *In Proceedings of the Tenth IASTED International Conference on Parallel and Distributed Computing and Systems* (PDCS'98), 1998, p. 541-546

[41] Benjamin Baran, Ruben Sosa, "A New Approach for AntNet routing", *In Proceedings of the Ninth International Conference on Computer Communications and Networks*, 2000

[42] Robert G. Gallager, " A Minimum Delay Routing Algorithm Using Distributed Computation", IEEE Trans. On Communications, vol. Com-25, No. 1, Jan 1977.

[43] Dimitri P. Bertsekas, Elim Gafni, "Projected Newton Methods and Optimization of Multicommodity Flows", IEEE Trans. On Automatic Control, Vol AC-28, No 12, Dec 1983

[44] Dimitri P. Bertsekas, Elim Gafni, Robert G. Gallager, "Second Derivative Algorithms for Minimum Delay Distributed Routing in Networks", IEEE Trans On Communications, Vol Com-32, No 8, Aug 1984

[45] John N. Tsitsiklis, Dimitri P. Bertsekas, "Distributed Asynchronous Optimal Routing in Data Networks", IEEE Trans. On Automatic Control, Vol AC- 31, NO.4. Apr, 1986

[46] Alexander Gersht, Alexander Shulman, "Optimal Routing in Circuit Switched Communication Networks", IEEE Trans. On Communications, Vol. 37, NO. 11, Nov, 1989

[47] John K. Antonio, Wei K. Tsai, and G. M. Huang, "Time Complexity of a Path Formulated Optimal Routing Algorithm", IEEE Trans. On Automatic Control, Vol. 39, No 9, Sep 1994.

[48] Wei K. Tsai, John K. Antonio, G.M. Huang, "Complexity of Gradient Projection Method for Optimal Routing in Data Networks", IEEE/ACM Trans. on Networking, Vol. 7, No. 6, Dec 1999

[49] Taehan Lee, Kyungsik Lee, Sungsoo Park, "Optimal Routing and Wavelength Assignment in WDM Ring Networks", IEEE Journal on Selected Areas in Communications, Vol. 18, No. 10, Oct 2000

[50] Srinivas Vutukury, J.J. Garcia-Luna-Aceves, "A Simple Approximation to Minimum-Delay Routing", In ACM SIGCOMM, pages 227-238, 1999

[51] Srinivas Vutukury, J.J. Garcia-Luna-Aceves, "A Traffic Engineering Approach based on Minimum-Delay Routing", Proc. IEEE IC3N 2000, Las Vegas, Nevada, USA, Oct 16-19, 2000

[52] Dimitri P. Bertsekas, Robert Gallager, "Data Networks", Prentice Hall, Englewood Cliffs, New Jersey, 1992.

[53] Eric J. Anderson, "A multicommodity Flow Based Approach to Adaptive Internet Routing", Ph.D thesis, CSE Dept, University of Washington.

[54] George B. Dantzig, Mukund N. Thapa, "Linear Programming", New York: Springer, 1997

[55] Dimitri P. Bertsekas, "Nonlinear Programming", Belmont, Mass.: Athena Scientific, 1999

[56] Gutjahr W. J. " A Graph-based Ant System and its convergence", Future Generation Computer Systems 16: 873-888, 2000

[57] Gutjahr W. J. " A generalized convergence result for the Graph-based Ant System", Technical Report 99-09, Dept. of Statistics and Decision Support Systems, University of Vienna, Austria: 1999

[58] Gutjahr W. "ACO Algorithms with Guaranteed Convergence to the optimal path", Information Processing letters 82: 145-153, 2002

[59] Thomas Stuzle and Marco Dorigo, " A Short Convergence Proof for a class of Ant Colony Optimization Algorithms", IEEE Transactions on Evolutionary Computation, VOL 6, NO 4, August 2002

[60] T. Stutzle and H. H. Hoos, " The MAX-MIN ant system and local search for the traveling salesman problem," in Proceedings of the 1997 IEEE International Conference on Evolutionary computation (ICEC 97), T. Back, Z. Michalewicz, and X. Yao, Eds. Piscataway, NJ: IEEE Press, 1997, pp. 309 – 314

[61] T. Stutzle and H. H. Hoos, "MAX-MIN ant system," Future Generation Computer System, Vol 16, NO 8, pp 889-914, 2000

[62] Bertsekas, D.P, Tsitsiklis, J.N. "Introduction to Probability", MIT *Lecture Notes*, 2000

[63] Mattuck, A, "Introduction to Analysis", Prentice Hall, New Jersey, 1999, p35

[64] Fraleigh, J.B, "A First Course in Abstract Algebra", Addison-Wesley: 1967

[65] Lagarias, J.C., J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions," SIAM Journal of Optimization, Vol. 9 Number 1, pp.112-147, 1998.

[66] Coleman, T.F. and Y. Li, "An Interior, Trust Region Approach for Nonlinear Minimization Subject to Bounds," SIAM Journal on Optimization, Vol. 6, pp. 418-445, 1996

[67] Coleman, T.F. and Y. Li, "On the Convergence of Reflective Newton Methods for Large-Scale Nonlinear Minimization Subject to Bounds," Mathematical Programming, Vol. 67, Number 2, pp. 189-224, 1994

[68] Tao Y., Hema T. K, Shivkumar K, Kenneth S. V. Saroj Y, " Optimization of OSPF Weights Using On-line Simulation", submitted, 2002

[69] RFC 768

[70] Peterson, L. L., & Davie, B. "Computer Networks: A System Approach", Morgan Kaufmann, 1996

[71] Danzig P.B., Liu Z., and Yan, L, "An evaluation of TCP Vegas by live emulation. Tech rep. UCS-CS-94, Computer Science Department, University of Southern California, Los Angeles, 1995

[72] James F. Kurose, Keith W. Ross, "Computer Networking, A Top-Down Approach Featuring the Internet", Addison Wesley, 2002.

[73] A.S. Nemirovsky, D.B.Yudin, "Problem Complexity and Method Efficiency in

Optimization", Wiley, N.Y., 1983

Appendices

Appendix 1 The Detailed Form of $\bar{P} = f_1(a,b)$ and $\bar{Q} = f_2(a,b)$.

The appendix lists the detailed form of $\bar{P} = f_1(a,b)$ and $\bar{Q} = f_2(a,b)$, the real solutions of systems of equation (5.20) and (5.21) that are computed by MATLAB.

$\bar{P} = f_1(a,b)$ =1/6/(-b+a*b+1-a)*(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)+2/3*(3*a^2*b-2*a^2+3*a*b^2-13*a*b+8*a-2*b^2+8*b-5)/(-b+a*b+1-a)/(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(

-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)-1/3*(-b+2*a-1)/(-b+a*b+1-a)

$\bar{Q} = f_2(a,b)$ =2*(1/6/(-b+a*b+1-a)*(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)+2/3*(3*a^2*b-2*a^2+3*a*b^2-13*a*b+8*a-2*b^2+8*b-5)/(-b+a*b+1-a)/(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a* b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)-1/3*(-b+2*a-1)/(-b+a*b+1-a))*(-1+b)/(-(1/6/(-b+a*b+1-a)*(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)+2/3*(3*a^2*b-2*a^2+3*a*b^2-13*a*b+8*a

-2*b^2+8*b-5)/(-b+a*b+1-a)/(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+3
6*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a
^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-1
2*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^
(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2
*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-
6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*
b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*
a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*
a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-1
2*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^
(1/2)*a)^(1/3)-1/3*(-b+2*a-1)/(-b+a*b+1-a))^2+(1/6/(-b+a*b+1-a)*(36*a^3*b-28*a^
3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-
48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3
+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*
a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+3
6*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^
3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-1
2*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*
b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2
-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3
+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*

a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)+2/3*(3*a^2*b-2*a^2+3*a*b^2-13*a*b+8*a-2*b^2+8*b-5)/(-b+a*b+1-a)/(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)-1/3*(-b+2*a-1)/(-b+a*b+1-a))^2*b+b*(1/6/(-b+a*b+1-a)*(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^

3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)+2/3*(3*a^2*b-2*a^2+3*a*b^2-13*a*b+8*a-2*b^2+8*b-5)/(-b+a*b+1-a)/(36*a^3*b-28*a^3-144*a^2*b^2+168*a^2*b-48*a^2+36*a*b^3+168*a*b^2-312*a*b+132*a-28*b^3-48*b^2+132*b-64-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a*b+12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)-12*(-12*a^4*b+9*a^4-36*a^3*b^2+36*a^3*b-6*a^3-36*a^2*b^3+378*a^2*b^2-450*a^2*b+141*a^2-12*b^4*a+36*a*b^3-450*a*b^2+606*a*b-216*a+9*b^4-6*b^3+141*b^2-216*b+84)^(1/2)*a)^(1/3)-1/3*(-b+2*a-1)/(-b+a*b+1-a))-1)

Appendix 2 To Validate Whether $f_1(a,b)$ is in the Domain of (0,1)

When $a \in (\frac{1}{1.1},1)$ and $b \in (\frac{1}{1.1},1)$, the figure of g(a,b) is plotted in Fig A.1.

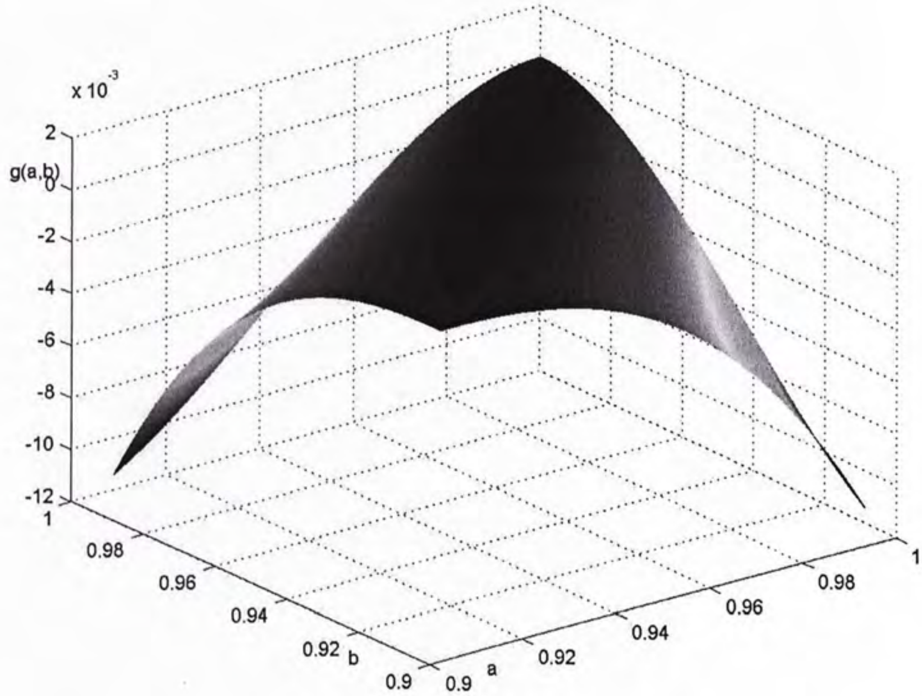

Fig A.1: The Figure of g(a,b).

From Fig A.1, it is obvious that Ω is the neighborhood of the point (0.9999, 0.9999). By using some MATLAB commands, it can be shown that $\Omega \subset \{(a,b) \mid a \in (0.9996,1), b \in (0.9996,1)\}$. We further evaluate the g(a,b) by using 10000 points in $\{(a,b) \mid a \in (0.9996,1), b \in (0.9996,1)\}$. There are totally 900 points of (a,b) that make $g(a,b) \geq 0$. Substitute these 900 points of (a,b) into $f_1(a,b)$, we have the minimum value of $f_1(a,b)$ is 2.9796e+005 that is much bigger than 1. So the results hold.